VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
STROJNÍ FAKULTA
KATEDRA ROBOTOTECHNIKY

Řízení manipulátoru

Control of the Robotics Manipulator

Student: Bc. Pavel Ehl
Vedoucí diplomové práce: doc. Dr. Ing. Petr Novák

Ostrava 2009

Obsah diplomové práce

Seznam použitých zkratek

Zkratka	Význam zkratky	Popis
BLDC motor	Brushless Direct Current motor	bezkartáčový stejnosměrný motor
MCU	Microcontroler Unit	jednotka mikrokontroléru
PWM	Pulse Wide Modulation	pulzní šířková modulace
EMC	Electronic Motion Controllers	řídicí jednotka pro pohony
qc	quadcount	Krok motoru (puls encodéru)
OM	Objekt manipulace	Předmět se kterým zařízení manipuluje
DI	Digital input	Logický vstup 0/1
DO	Digital output	Logický výstup 0/1

1. Úvod

Během posledních let došlo v oblasti automatizace k výraznému pokroku, a to především v řízení automatizovaných systémů. Pod tlakem konkurence vznikají nové a nové nápady a technologie pro zvyšování produkce a rychlosti výroby. Za pomoci těchto technologií bylo a je možné stále zvyšovat výrobní výkony a kvalitu výrobků, a současně snižovat náklady a dobu výrobního cyklu.

Mechatronika v konstrukci a stavbě strojů vede k tomu, že software zabírá stále vyšší podíl na dalším vývoji a na splnění požadovaných funkcí strojů a zařízení. V těchto systémech mají klíčovou funkci elektronicky řízené pohony. [1]

Elektronicky řízené pohony konají programové příkazy vložené do řídicího systému pohonu. Tím se stávají součástí mechatronického systému. Každý elektronicky regulovaný pohon ve stroji nebo zařízení tak představuje určitý mechatronický subsystém. [1]

Tyto řídicí systémy předpokládají pro pohony používání řídicích jednotek Electronic Motion Controllers (EMC).

Mezi jejich základní funkce patří také možnost nastavení vlastního rychlostního profilu pohonu. Tyto jednotky dovolují nastavit jakýkoliv potřebný rychlostní profil dle potřeb uživatele. Je možné nastavovat rychlosti pohonu plynule bez rázů, a tím snížit jednak namáhání součástí, a jednak potřebný moment a velikost pohonu. Tímto způsobem se dosahuje snížení energetické náročnosti zařízení.

Každý regulovaný pohon ve výrobě plní svůj konkrétní úkol, jako např. nastavení polohy, kontinuální posuv materiálu, nebo navíjení a odvíjení svitků. Tato činnost je v praxi realizována kooperací softwarové (řídicí program) a hardwarové části (frekvenční měnič a pohon).

Mechanická energie motoru je přenášena pomocí převodů, vřeten, ozubených řemenů nebo hřídelí do výrobního procesu.

Funkce pro řízení pohybu jsou systematicky popsány a standardizovány jen v několika málo strojírenských oborech, například ve stavbě obráběcích strojů. V jiných odvětvích průmyslu jsou řešeny spíše řemeslně případ od případu. Využitím vhodného stavebnicového systému, který obsahuje také softwarové prvky pro řízení pohybu, a efektivní metodiky návrhu je možné náklady na realizaci určité funkce stroje významně zredukovat.[1]

Software pro řízení pohybů a tím také pro vyřešení úkolů pohonu lze v principu realizovat buď v centrální řídicí jednotce, nebo decentralizovaně přímo v pohonu. V některých případech je vhodnější, aby pohon pracoval a řídil se sám a zmenšil tím zatížení centrálního řídicího systému, který se snadno zvládnutelnou jednotkou PLC vystačí pro velký počet řízených pohonů.

V souladu s tím se masivně rozšířilo i programové řízení motorů pomocí výkonných mikroprocesorů (MCU) využívajících pulzní řízení založené na pulzní šířkové modulaci (PWM). Pulzní šířková modulace se realizuje u motorů řízením doby sepnutí (připojení) konstantního stejnosměrného napájecího napětí. Je to jedna z nejlepších a univerzálních metod, kterou lze zároveň regulovat stejnosměrné i střídavé motory, a je snadno hardwarově i softwarově realizovatelná přímo na čipu mikrokontroléru.

Předmětem této diplomové práce je uvedení manipulátoru, zkonstruovaného na katedře robototechniky VŠB-TU Ostrava, do provozu. Jedná o mechatronický systém se čtyřmi řídicími jednotkami Maxon EPOS 70/10 a pohony Maxon, řízené pomocí PWM. Manipulátor bude řízen softwarem, který je součástí této diplomové práce.

2. Současný stav mechatronických systémů

2.1 Druhy používaných pohonů

Od doby kdy byl objeven první elektromotor, bylo vynalezeno a vyzkoušeno několik různých druhů elektromotorů. Některé se již nepoužívají, jiné se začali používat teprve nedávno.

Používané druhy elektromotorů

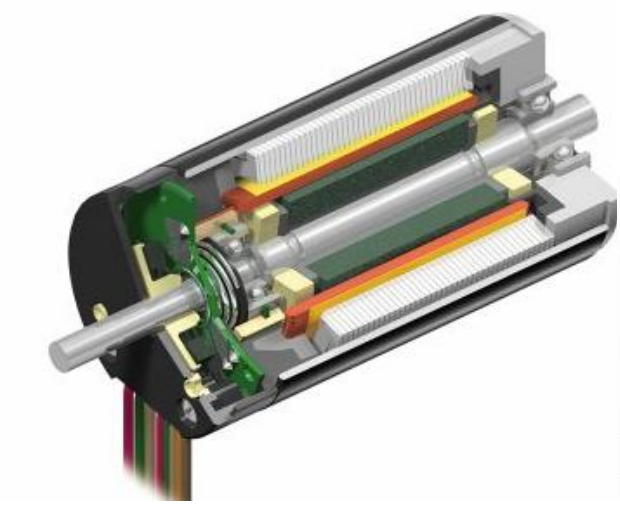
- Kartáčový (komutátorový) stejnosměrný motor (Brushed DC Motor)
- Univerzální motor (na střídavé i stejnosměrné napětí)

- Bezkartáčový stejnosměrný motor (BLDC Motor)
- Střídavý 3-fázový indukční (asynchronní) a synchronní motor (AC Motor)
- Střídavý jednofázový motor (AC Motor)
- Krokový bipolární a unipolární motor (Stepper Motor)
- Spínaný reakční motor (Switched Reluctance Motor = SR Motor)

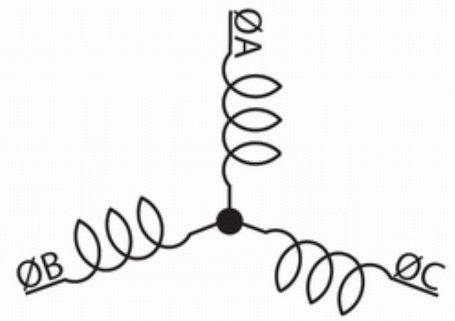
Manipulátor je osazen čtyřmi elektromotory MAXON, z nichž 3 jsou klasické stejnosměrné kartáčové (DC) motory o výkonech 2 x 70W a 80W a jeden bezkartáčový (BLDC) motor o výkonu 400W.

2.2 BLDC motor

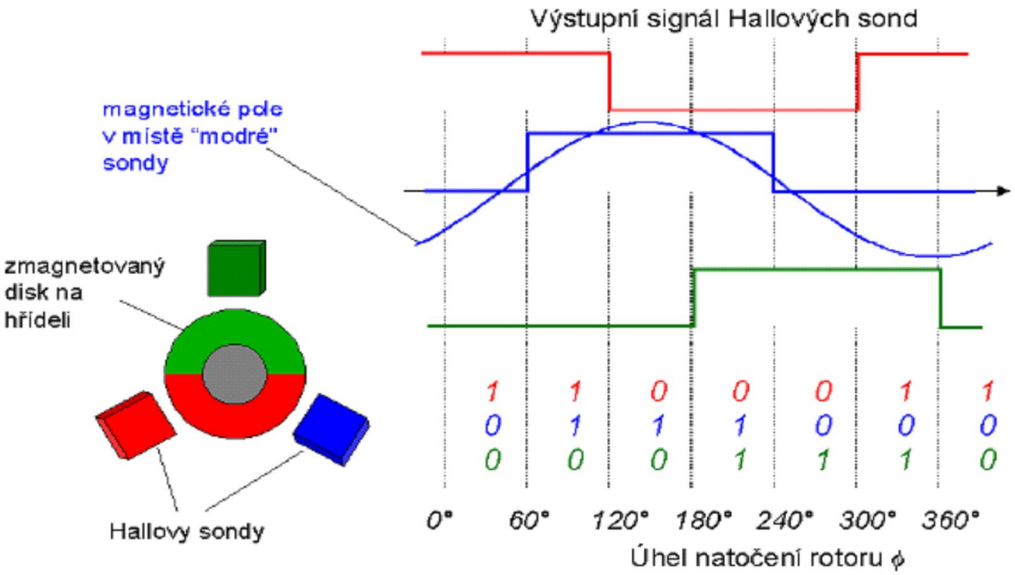
Bezkartáčový BLDC motor (Brushless DC motor) patří do kategorie stejnosměrných motorů, i když je strukturou podobný střídavému 3-fázovému synchronnímu motoru. Z tohoto důvodu nelze na motor připojit přímo stejnosměrné napětí ze zdroje, ale je nutné provádět jeho postupné spínání. Stator je běžně tvořen 3 budícími vinutími zapojenými do hvězdy (obr.2), i když menší motory někdy mají jen 2. Rotor je tvořen permanentními magnety a může být v provedení vnitřním, vnějším nebo paralelně se statorem. Tento typ motoru je nutné řídit speciální řídicí jednotkou, která musí být schopna generovat sinusový signál pro buzení točivého elektrického pole ve statoru motoru (obr.3).



Obr.1 EC motor [18]



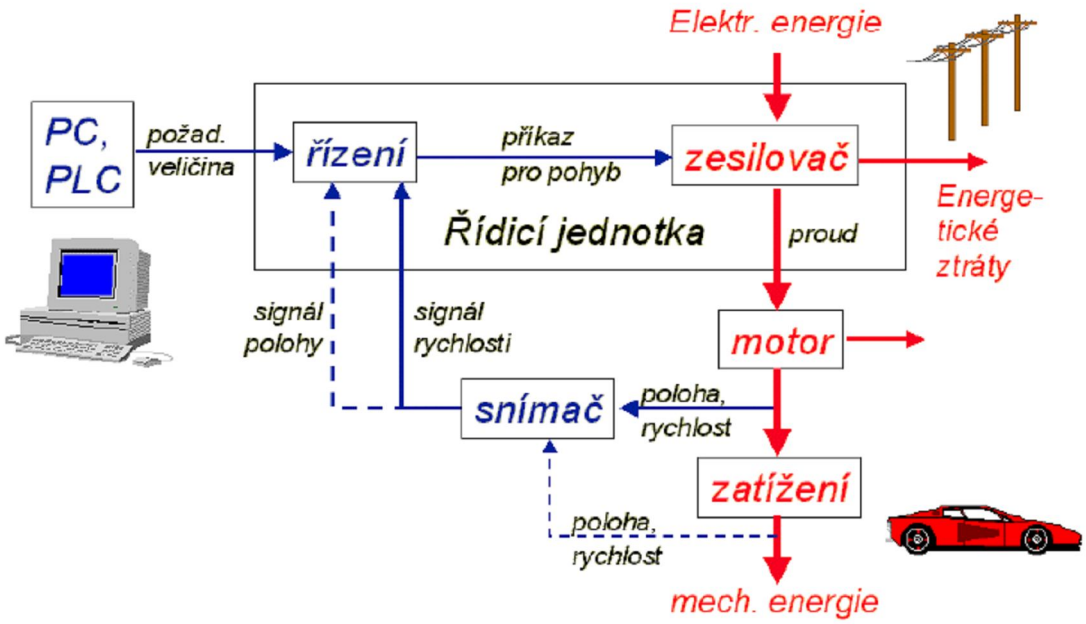
Obr.2 Zapojení vinutí EC motoru



Obr.3 Průběh komutace napětí v Hallových sondách pro buzení EC motoru[12]

2.3 Řízení pohonů

Schéma řízení pohonů:



Obr.4 Schéma regulační smyčky pohonů [11]

Řízení BLDC motoru:

Proud vinutími je sekvencován, přičemž 3-fázový průběh se simuluje současným napájením vždy jen 2 vinutí, každé opačným směrem proudu (jedno kladným, druhé záporným) nebo celé napětí k jednomu a 1/2 napětí na zbylá dvě vinutí. Tyto hodnoty bývají označovány jako HIGH a LOW signál. (kombinace HHL odpovídá vrcholu sinusoidy na příslušném vinutí, zatímco HLL odpovídá středu). Rychlost otáčení se řídí frekvencí (časováním) sekvencování, zatímco střídou a frekvencí impulsů uvnitř sekvencí se určuje "hladkost" běhu. Točivý moment je definován velikostí proudu, přičemž vysokonapěťové impulsy vytvářejí stejný efekt. [3]

Řídicí jednotka udržuje rychlost a mění ji podle velikosti vstupního řídicího signálu a urychluje i brzdí motor s potřebným průběhem rychlosti.

Řízení stejnosměrných DC motorů:

Rychlost motorů DC s komutátorem je v jednoduchých aplikacích možno nastavit velikostí napájecího (stejnosměrného) napětí. Rychlost motoru bez zatížení je přímo úměrná napětí. Každý Nm zatížení ji sníží o konstantní hodnotu. Připojením napětí se motor rozběhne s časovou konstantou na rychlost odpovídající zatížení. Rychlé zastavení motoru s touto časovou konstantou se provádí zkratováním vinutí. Informace pro ovládání v tomto případě může pocházet od koncových spínačů.

Pro dokonalejší ovládání rychlosti motorů DC se použije některá z řídicích jednotek rychlosti s čtyřkvadrantovým řízením (např. MAXON EPOS).

Druhy řízení 3-fázových střídavých a BLDC elektromotorů:

- Jednoduché sekvencování pulzů
- Lichoběžníkové prostorové vektorové řízení (Trapezoid Space vector)
- Sinusové prostorově vektorové řízení
- FOC řízení

3. Řídicí jednotky MAXON - EPOS

Jednotky podporují tyto módy řízení podle druhu snímače otáčení pro zpětnou vazbu:

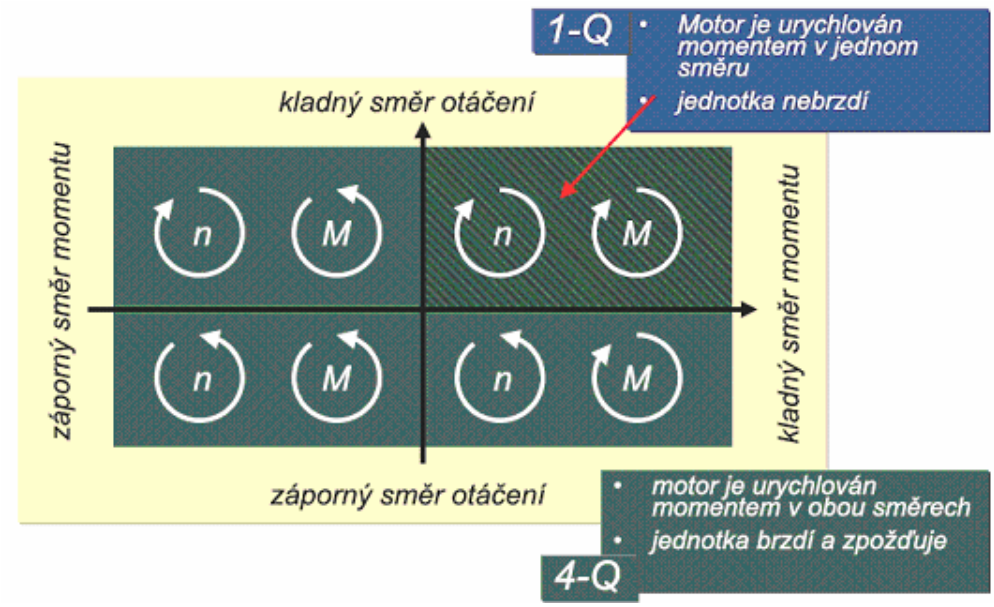
- mód s inkrementálním snímačem
- tachodynamem
- indukovaným napětím ve vinutí motoru.
- hallovými sondami integrovanými v některých motorech MAXON

Každý EC motor (BLDC) musí být napájen jednotkou, která vytváří elektronickou komutaci.

Všechny jednotky EPOS obsahují regulátor rychlosti s možností nastavit:

- rozběhovou rampu
- brzdicí rampu
- požadovanou rychlost

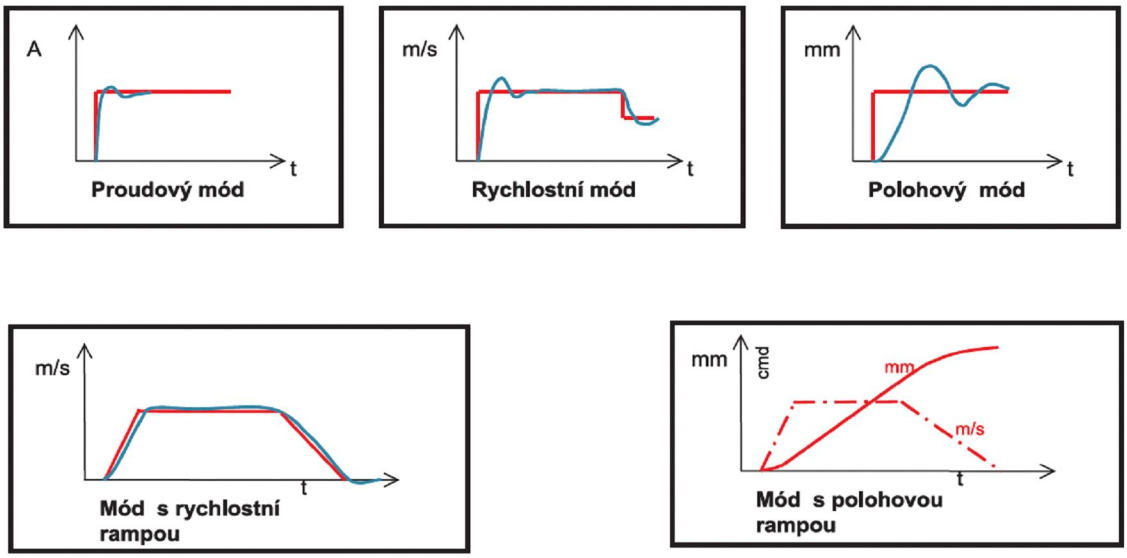
Řídicí signál a nastavení parametrů regulace mohou být analogové nebo digitální ve formě signálu RS232. Lze volit jednotku pro jednokvadrantové nebo čtyřkvadrantové řízení.



Obr.5 Schéma čtyřkvadrantového řízení pohonu [11]

3.1 Módy řízení jednotek EPOS

- **Polohový mód**
- **Rychlostní mód**
- **Proudový mód (momentový)**



Obr.6 Módy řízení EPOS [11]

Polohové řízení pohonu:

V případě jednoduchých aplikací není natolik nežádoucí menší překmit pohonu v koncové poloze. Pro takovéto případy postačí jednodušší a levnější jednotky pro rychlostní řízení pohonu nebo ještě jednodušším způsobem pouze nastavením napětí zdroje a koncovým spínačem. Motory MAXON mají velmi krátkou časovou konstantu (asi 10 ms) a zastaví poměrně rychle.

U složitějších aplikací použijeme řídicí jednotky s regulátorem polohy se zpětnou vazbou. Řídicí jednotky MAXON pro polohové řízení mají pro EC motory přednastaven sinusový průběh proudu.

Řídicí jednotky MAXON MIP pro DC nebo EC motory jsou schopny řídit pohon z požadavků na:

- Koncovou polohu
- Rozběhovou rampu

- Brzdící rampu
- Rychlost pohybu

Ve většině případů postačí řízení rychlosti se zpětnou vazbou na zabudované Hallovy sondy. Hustota signálů ze sond je poměrně nízká. Hallovy sondy generují 6 hran na průchod jednoho páru pólů. To stačí pro elektronické řízení komutace i při dynamickém rozběhu a na stabilní řízení v oblasti vyšších otáček. U dvoupólového motoru je tato oblast přibližně nad 1000 ot/min^{-1}.

V jednotce se nastaví požadované zesílení PID regulátoru pro polohu a PI regulátoru pro rychlost a proud. Hodnoty se vloží do jednotky pomocí PC nebo IPC a je možné je i během provozu podle potřeby měnit. Pohon musí být doplněn IRC snímačem pro zpětnou vazbu. Naladění hodnot regulátorů se provádí přes aplikaci „EPOS Studio“, případně „EPOS User Interface“, ve kterém se spustí “Autotuning „ regulátorů jednotky EPOS. Jednotka si nastaví hodnoty složek regulátorů podle aktuální zátěže pohonu.

3.2 Vyráběné řídicí jednotky EPOS

Všechny řídicí jednotky EPOS jsou vybaveny sériovou komunikační linkou přes RS232. Nová řada řídicích jednotek polohy EPOS umožňuje komunikaci s často používanou sběrnicí CANopen. Použití jednotky EPOS v módu řízení rychlosti nebo řízení proudu přináší možnost digitálních vstupů, požadované hodnoty rychlosti, proudu, rozběhových ramp a omezení proudu v dynamickém režimu. Při řízení jednoho motoru komunikuje i v RS232. V roce 2004 začala výroba typu EPOS 24/1 s napájením 24 V a výstupem 1 A trvale, 2 A krátkodobě a silnějšího typu EPOS 24/5 s napájením 24 V a výstupem 5 A trvale, 10 A krátkodobě. V současné době běží výroba typu EPOS 70/10 s napájením do 70 V a výstupem 10 A trvale a 25 A krátkodobě.

Následníkem těchto modelů jsou vylepšené verze EPOS2 50/5, které poskytují uživateli možnost připojení přes USB. Poslední novinkou jsou programovatelné jednotky EPOS P, které mají vlastní paměť pro programový kód, a mohou tak pracovat nezávisle na PC či PLC a jsou schopny řídit další jednotky.

Obr.7 EPOS 24/1 [13]

Obr.8 EPOS2 50/5 [13]

4. Požadavkový list

Proveďte návrh řídicího systému pro stávající angulární robot se 4 osami realizovaný na katedře robototechniky VŠB-TU Ostrava.

Hardwarová část:

- Vyřešit napájecí zdroj pro řídicí systém a pohony + jeho krytování
- Navrhnout mechanické dorazy pohybu jednotlivých os
- Navrhnout koncové spínače jednotlivých os
- Vyřešit vedení energií a signálů k pohonům
- Použití následujících komponent:

  Motory MAXON: EC 60
  RE 36
  F 2260

  Mikrokontroléry MAXON: EPOS 70/10
- Provést elektrické zapojení mikrokontrolérů a motorů
- Zvážit možnosti osazení ramen inklinometrem nebo akcelerometrem s analogovým výstupem pro usnadnění počáteční inicializace
- Dokumentace elektrického zapojení manipulátoru

Softwarová část:

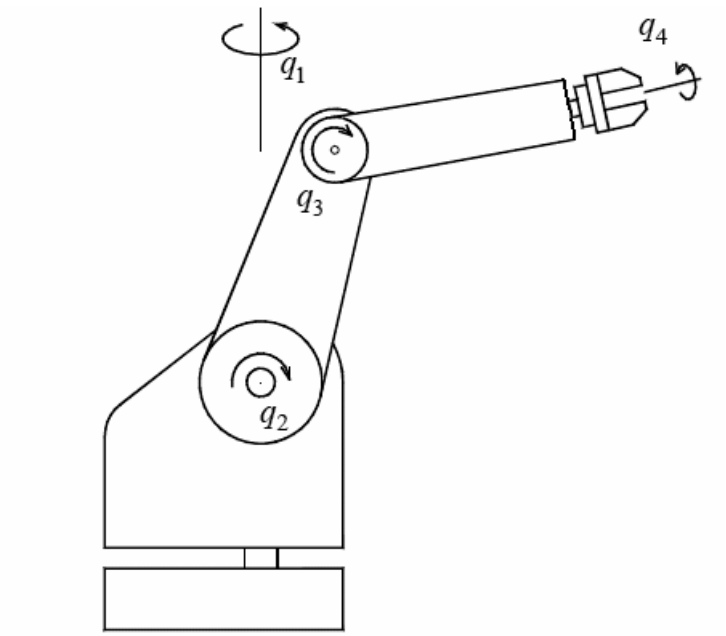
- Vytvořit základní rutiny pro ovládání jednotlivých os
- Navrhnout základní interpolace s ohledem na kinematickou strukturu
- Navrhnout robustní systém inicializace
- Módy řízení robotu:

  Polohové

  Rychlostní

5. Použité zařízení

5.1 Manipulátor



Obr.9 Kinematická struktura manipulátoru

Počet stupňů volnosti	-	4 rotace
Nosnost	-	3 kg
Hmotnost	-	36,7 kg
Dosah ramena od osy základu	-	964 mm

Rozsahy pohybů:

Osa č.	Popis	Rozsah pohybu	Max. rychlost
1	Vertikální rotace v základu	330°	12,6 min^{-1}
2	Naklápění ramena 1	± 73°	17 min^{-1}
3	Naklápění ramena 2	330°	19,3 min^{-1}
4	Rotace příruby	360°	25,2 min^{-1}

Tab.1 Rozsahy a rychlosti pohybů manipulátoru

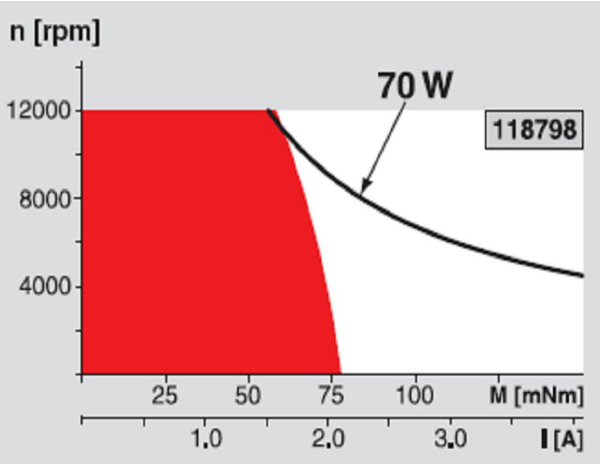
5.2 Rotační osa (rotace základu, rotace příruby)

Motor – Maxon RE36 (DC motor)

Nominální napětí	24 V
Točivý moment	78,2 mNm
Nominální otáčky	5530 min^{-1}
Maximální otáčky	12000 min^{-1}
Otáčky naprázdno	6210 min^{-1}
Výkon	70 W
Rozběhový moment	783 mNm
Proud naprázdno	105 mA
Nominální proud	2,25 A
Startovací proud	21,5 A
Účinnost	85 %
Mechanická konstanta	5,89 ms
Moment setrvačnosti rotoru	69,9 g/cm^2

Obr.10 DC motor RE36 [18]

Tab.2 Parametry motoru Maxon RE36 [15]



Obr.11 Charakteristika motoru RE36 [15]

Obr.12 Převodovka Maxon [19]

Planetová převodovka GP 32 C

Převodový poměr	246:1
Počet stupňů	4
Maximální vstupní otáčky	8000 min^{-1}
Maximální výstupní moment	6 Nm
Materiál převodů	keramika

Tab.3 Parametry převodovky Maxon GP32C [23]

5.3 Rotační osa 2 (sklápění prvního ramene)

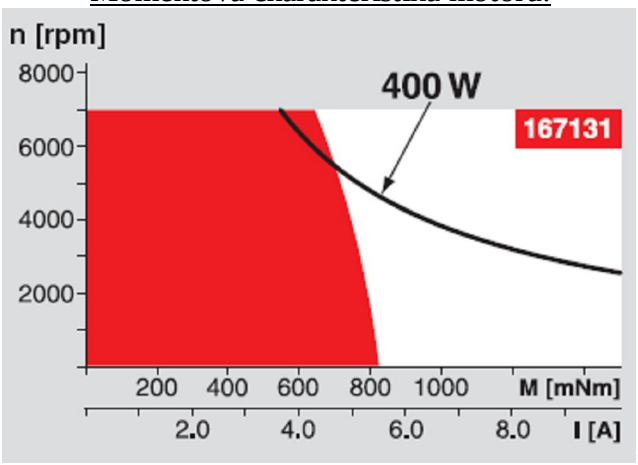
Motor – Maxon EC60 (BLDC motor)

Nominální napětí	48 V
Točivý moment	747 - 830 mNm
Nominální otáčky	2680-4960 min^{-1}
Maximální otáčky	7000 min^{-1}
Otáčky naprázdno	5370 min^{-1}
Výkon	400 W
Rozběhový moment	11800 mNm
Proud naprázdno	733 mA
Nominální proud	9.38 A
Startovací proud	139 A
Mechanická konstanta	3.81 ms
Moment setrvačnosti rotoru	831 g/cm^2

Tab.4 Parametry motoru Maxon EC60 [17]

Obr.13 Motor MAXON EC [22]

Momentová charakteristika motoru:



Obr.14 Charakteristika motoru EC60 [17]

Planetová převodovka GP 81

Převodový poměr	92,7
Počet stupňů	3
Maximální vstupní otáčky	3000 min^{-1}
Maximální výstupní moment	120 Nm
Materiál převodů	Ocel

Tab.5 Parametry převodovky Maxon GP81[20]

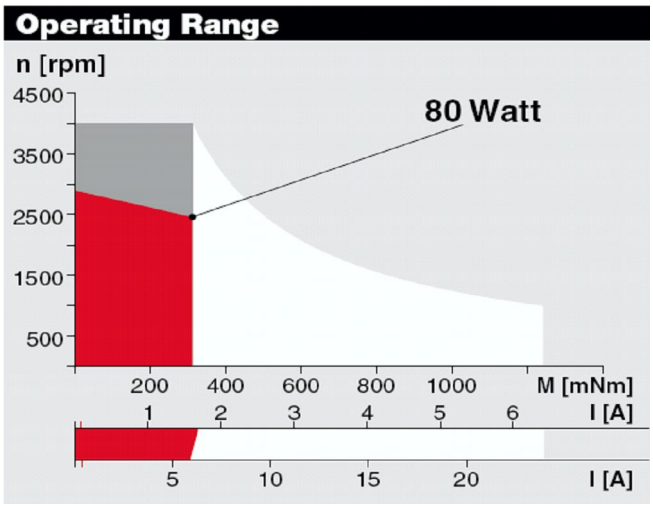
5.4 Rotační osa 3 (sklápění druhého ramene)

Motor – Maxon F2260 (feritové magnety)

Výkon	80 W
Nominální napětí	24 V
Otáčky naprázdno	2230 min^{-1}
Moment zastavení motoru	1670 mNm
Proud naprázdno	168 mA
Startovací proud	16,7 A
Maximální otáčky	4000 min^{-1}
Rozběhový moment	16,7 A
Maximální trvalý proud	3,21 A
Maximální trvalý moment	321 mNm
Účinnost	79 %
Mechanická konstanta	5,89 ms
Moment setrvačnosti rotoru	69,9 g/cm^2

Obr.15 DC MAXON F 2260 [24]

Tab.6 Parametry motoru Maxon F2260 [16]



Obr.16 Charakteristika motoru F 2260 [16]

Planetová převodovka GP 62 A

Převodový poměr	139:1
Počet stupňů	3
Maximální vstupní otáčky	8000 min^{-1}
Maximální výstupní moment	50 Nm
Materiál převodů	keramika

Tab.7 Parametry převodovky Maxon GP62A [21]

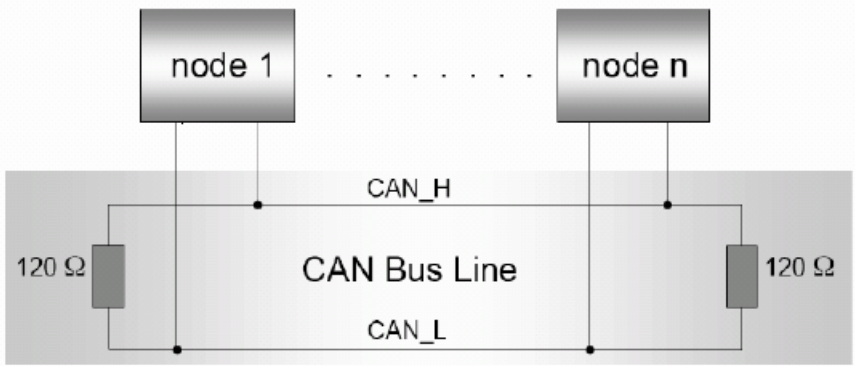
5.5 Jednotka EPOS 70/10

Obr.17 EPOS 70/10 [7]

ELEKTRICKÁ DATA	
Napájecí napětí Vcc (zvlnění < 10 %)	11 - 70 V
Max. výstupní napětí	0,9 x Vcc
Max. výstupní proud I_{max} (< 1 s)	25 A
Výstupní proud trvalý I_{cont}	10 A
Vzorkovací frekvence proudového PI regulátoru	10 kHz
Vzorkovací frekvence rychlostního PI regulátoru	1 kHz
Vzorkovací frekvence polohového PID regulátoru	1 kHz
Max. rychlost (2-pólové motory)	25000 min-1
Zabudovaná tlumivka	25 µH / 10 A
VSTUPY	
Analogové vstupy	2 analog. vstupy 10-bit rozlišení, 0 ... +5 V
Digitální vstupy	8 dig. vstupů
Signály inkrementálního snímače	A, A', B, B', I, I' (max. 1 MHz)
Signály Hallových sond	H1, H2, H3
CAN-ID (identifikace CAN uzlu)	DIP přepínačem 1 ... 7
VÝSTUPY	
Digitální výstupy	4 dig. výstupy
INTERFACE	
RS232	RxD; TxD (max. 115 200 bit/s)
CAN	high; low (max. 1 Mbit/s)

Tab.8 Parametry jednotky EPOS 70/10 [7]

Zapojení všech modulů přes sběrnici CANopen:



Obr.18 Zapojení sběrnice CAN [14]

5.6 Brzda AB41 motoru EC60

PARAMETR	HODNOTA
Statický brzdný moment	> 2,0 Nm
Moment setrvačnosti	45 gcm^2
Maximální otáčky	10000 min-1
Váha	0,19 kg
Jmenovité napětí	24 VDC
Odpor	R = 72 Ohm
Proud	333 mA
Doba zapnutí	100 %
Čas přitažení	2 ms
Čas odpojení	25 ms

Tab.9 Parametry brzdy AB41 [6]

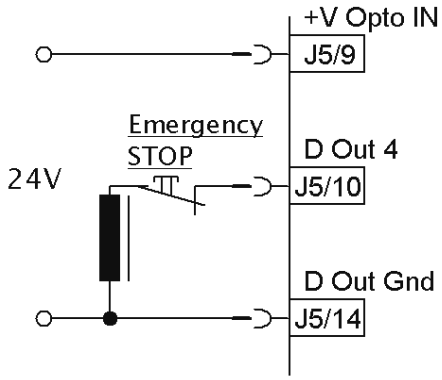
Obr.19 Brzda AB41[6]

Brzda AB41 stejnosměrná brzda s permanentním magnetem. Je zabrzděná ve stavu bez připojeného napájení. Brzda je určena pouze pro držení klidové polohy nebo při odpojení napájení motoru. Není vhodná pro brzdění otáčející se hřídele motoru. Doporučuje se snížit napájecí napětí brzdy na 12V poté, co byla nabuzena, aby se snížily tepelné ztráty. [6]

Zatížitelnost digitálního výstupu č.4, na který je brzda připojena, je 500 mA, což je pro brzdu postačující.

Brzda je napájena 24V / 1A spínaným zdrojem umístěným v rozvodné skříni společně se zdroji pro motory.

Připojení Brzdy motoru EC60:



Brzda je připojena na jednotku, která řídí motor EC60 tzn. EPOS #2, a je připojena podle obrázku na zdroj 24V. Příkon (spotřeba) brzdy je 8W, tedy 333mA při 24V.

Obr.20 Zapojení brzdy AB41 k EPOS 70/10

5.7 Koncové spínače

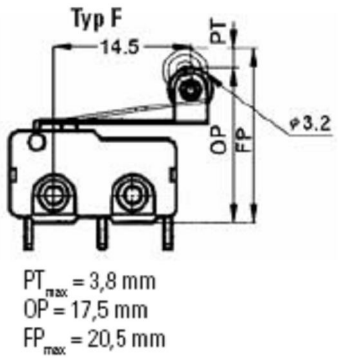
Pro každou osu je k jednotkám EPOS nutné připojit koncové spínače. Jsou použity mechanické mikrospínače s kladičkou P-B175-F (250V / 5A). Mikrospínače jsou napájeny ze zdroje 24V a jejich signály jsou přivedeny na digitální vstupy příslušných jednotek EPOS dle tabulky 10.

Připojení mikrospínačů:

Název spínače	**Význam**	**Připojení**
Positive limit switch (DI5)	Koncová poloha v kladném směru rotace	J5/4
Negative limit switch (DI6)	Koncová poloha v záporném směru rotace	J5/3
Common signal 2	Záporná větev napájení -24V	J5/1,2,14

Tab.10 Zapojení koncových spínačů k EPOS

Použité taktilní mikrospínače

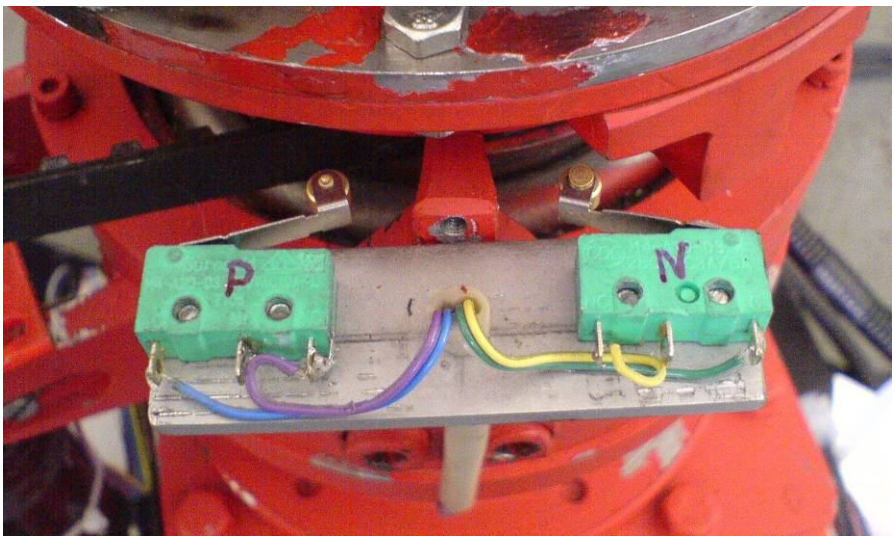


Obr.21 Použité mikrospínače pro koncové spínače [25]

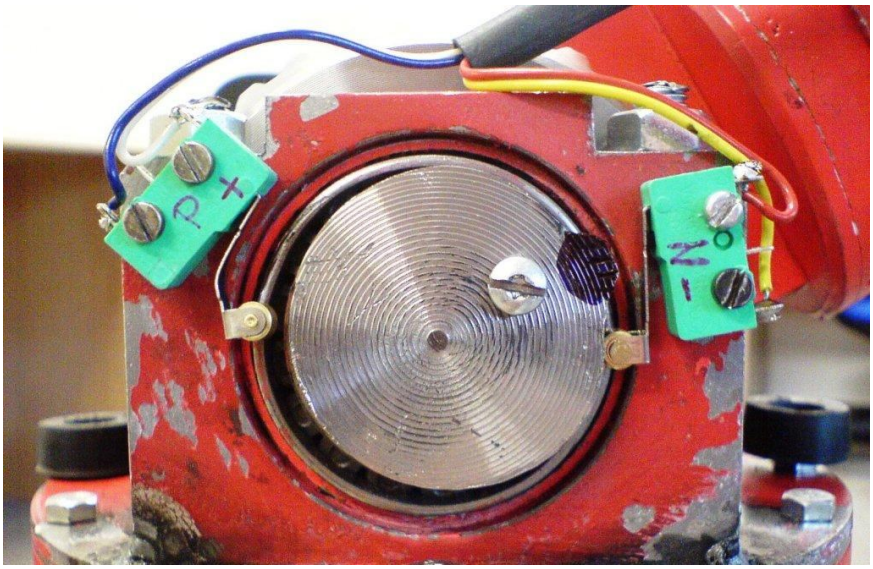
Parametry signálu z koncových spínačů

Vstupní napětí	24V
Proud mikrospínačem	13 mA
Zátěž na zdroj (6 spínačů na 3 EPOS jednotkách)	78 mA

Tab.11 Odběr koncových spínačů



Obr.22 Instalované koncové spínače osy 1



Obr.23 Instalované koncové spínače osy 2

Koncové spínače osy 3 jsou spínány radiální vačkou připevněnou na hřídel kuželového soukolí, které pohání rameno 2 v poměru 1:1.

6. Návrh napájecího zdroje pro pohony robotu

Napětí zdroje:

Vzhledem k použití motoru MAXON EC60, který má jmenovité napájecí napětí 48 V a jednotky EPOS 70/10, je třeba volit zdroj o velikosti výstupního napětí dle výpočtu uvedeném v katalogovém listu EPOS70/10 [7] :

$$V_{CC} = \frac{U_N}{n_0} * \left(n_B + \frac{\Delta n}{\Delta M} * M_B \right) * \frac{1}{0,9} + 1 \qquad (1)$$

$$V_{CC} = \frac{48}{5370} * (5000 + 0,457 * 700) * \frac{1}{0,9} + 1$$

$$\underline{\underline{V_{CC} = 50,516\,V}}$$

U_N …… Nominální napětí motoru [V]

n_0……Otáčky motoru na prázdno [min^{-1}]

n_B …… . Pracovní otáčky motoru [min^{-1}]

$\Delta n/\Delta M$… Gradient motoru [min * mNm^{-1}]

M_B …… Pracovní moment motoru [mNm]

Výkon zdroje:

Zdroj musí mít potřebný výkon pro napájení všech 4 pohonů:

$P = P_1 + P_2 + P_3 + P_4$ (2)

P = 400 + 80 + 70 + 70

<u>P = 620 W</u>

Pro napájení pohonů robotu byla zvolena dvojice spínaných zdrojů firmy MeanWell PSP-500-48, které budou pracovat v paralelním zapojení. Celkový výkon zdrojů bude cca 1000 W z důvodu rezervy pro rozběh motorů. Dle katalogového listu jsou navíc zdroje schopny krátkodobého přetížení až 125%, což odpovídá 1250 W. Měly by tak být schopny pokrýt z části i rozběhové proudy motorů.

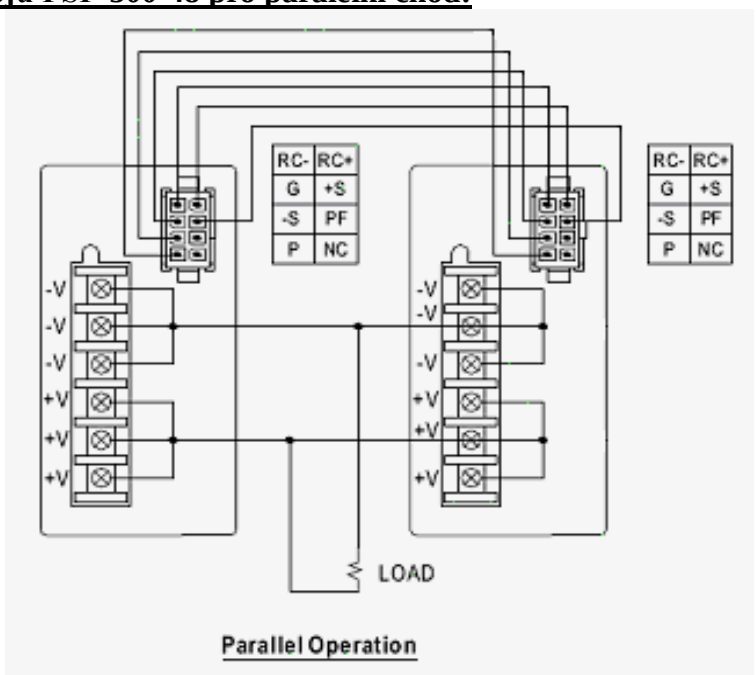
Charakteristiky zdroje PSP-500-48 [27]:

Výstupní napětí:	48 V (regulovatelné od 41 V do 58 V – nastaveno na 58V)
Výstupní proud:	10,5 A (krátkodobě až 13,125 A)
Výstupní výkon:	504 W (krátkodobě až 630W)
Vstupní napětí:	90 – 264 V
Vstupní proud:	3,5 A
Účinnost:	86 %
Ovládací signál:	0 –0,8 V (On) , 4 - 10 V (Off)
Rozměry:	278 * 129 * 63.5mm (D*Š*V)

Obr.24 Zdroj PSP-500-48 pro pohony [27]

Obr.25 Zdroj 24V/1A pro koncové spínače a brzdu [26]

Zapojení zdrojů PSP-500-48 pro paralelní chod:

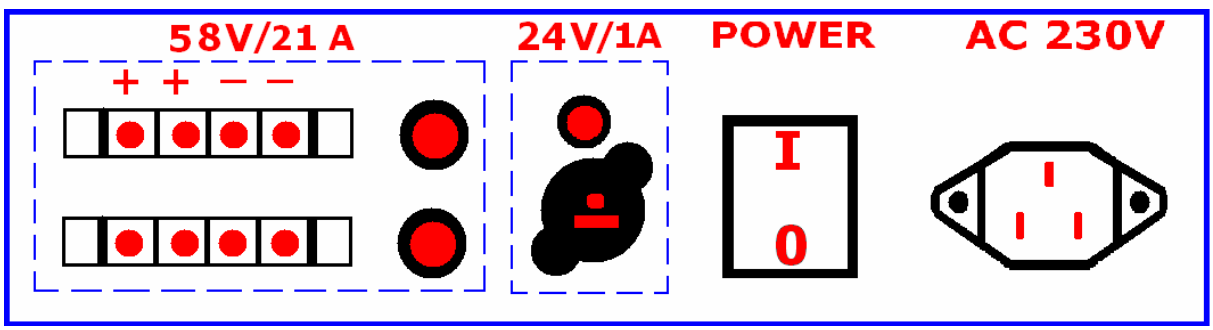


Obr.26 Paralelní zapojení zdrojů PSP-500-48 [27]

Pro zdroje byla použita univerzální montážní přístrojová skříň firmy ABB 275x570x140 mm, do které byly umístěny všechny potřebné zdroje napětí pro manipulátor: 2x504W/58V a 24W/24V. Skříň byla osazena tak, aby byla snadná možnost připojení jak napájení zdrojů, tak i výstup. Zdroje 58V jsou propojeny a pracují v paralelním režimu, takže jsou schopny dodávat trvale až 21 A a krátkodobě až 26,25 A. Skříň se zdroji byla opatřena větracími otvory pro lepší cirkulaci vzduchu a chlazení zdrojů.

Obr.27 Elektroinstalační box pro zdroje [28]

Popis panelu zdrojové skříně:

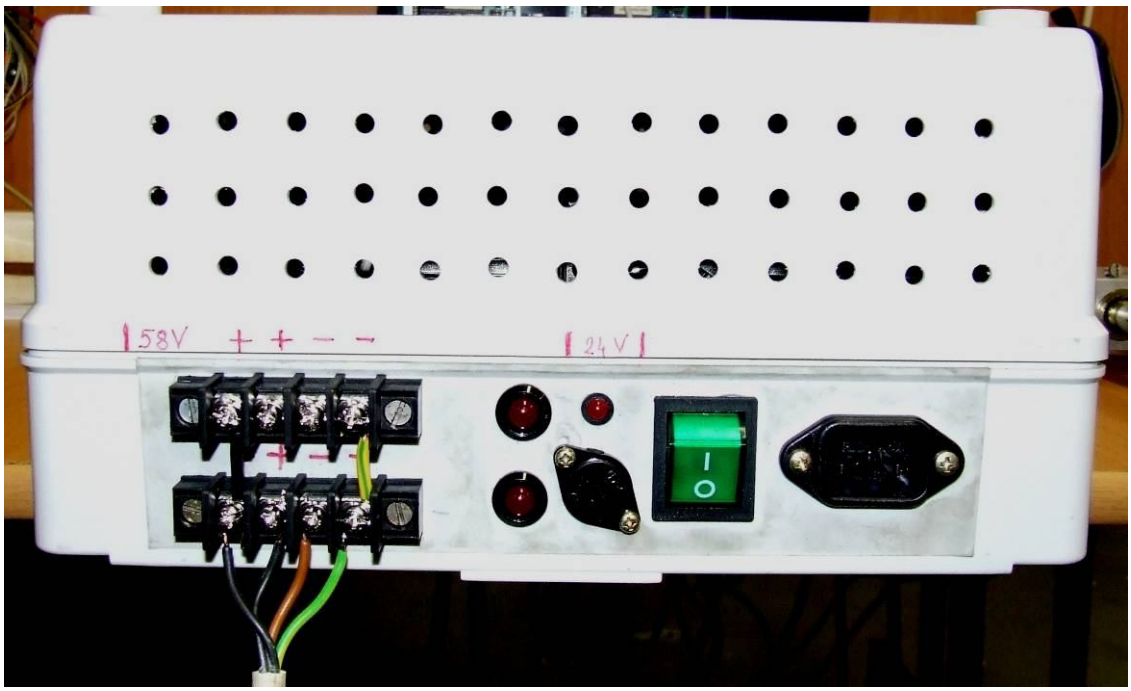


Obr.28 Schéma předního panelu zdroje

Zapojení předního panelu:

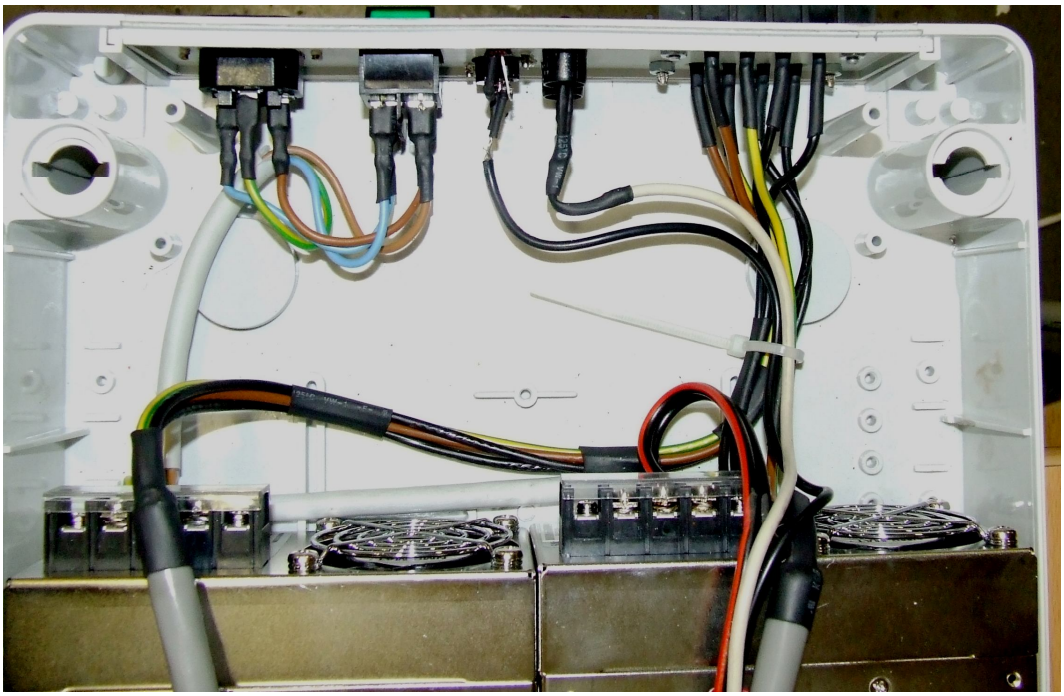
Čelní panel skříně je osazen ISO konektorem pro připojení napájení pomocí běžného napájecího kabelu. Dále obsahuje podsvícený kolébkový síťový vypínač pro zapínání a vypínání napájení 230V. Výstupní část obsahuje konektor pro připojení napětí pro brzdu a koncové spínače spolu se signalizační LED diodou. LED dioda je doplněna rezistorem 1kΩ a LED diody pro zdroje 58V jsou s odpory 3kΩ. Dále panel obsahuje dvě šroubové svorkovnice pro připojení napětí 58V pro motory. Každá svorkovnice obsahuje 2 svorky s kladným napětím a 2 se záporným. Každý zdroj 58V má vlastní svorkovnici a kontrolní LED diodu. Propojení výstupů zdrojů je provedeno propojkami na výstupních svorkovnicích.

Čelní panel zdroje:



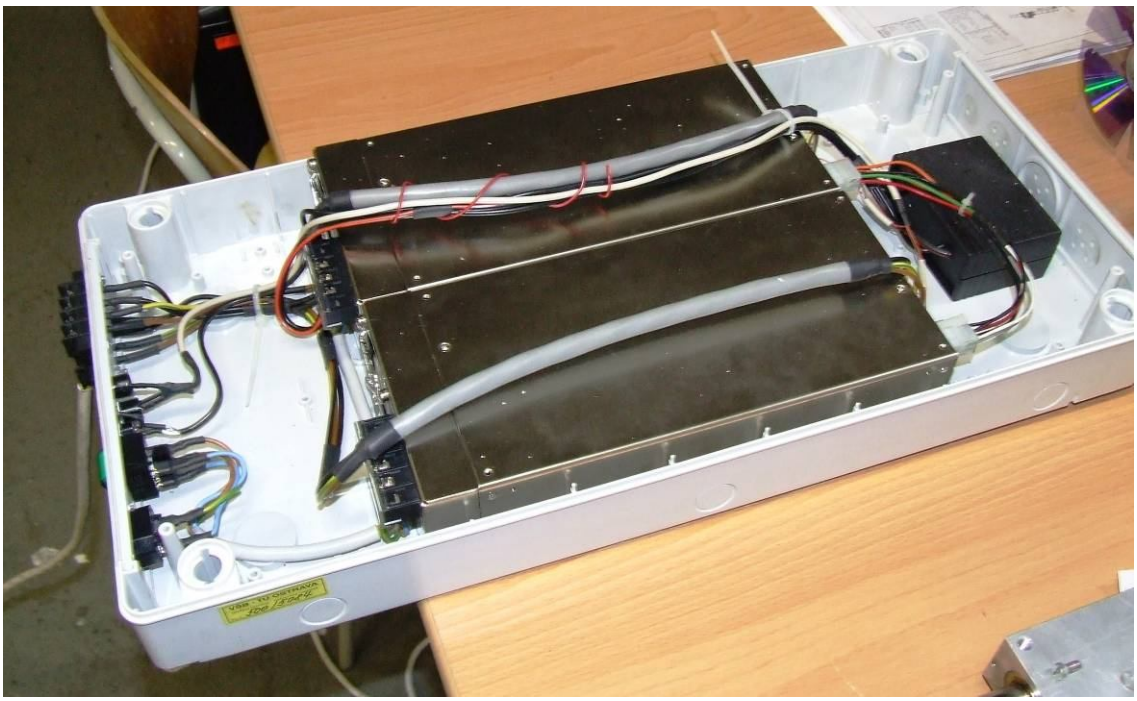
Obr.29 Přední panel zdroje

Vnitřní zapojení čelního panelu:



Obr.30 Zapojení předního panelu zdroje

Vnitřní uspořádání zdrojů zdrojové skříně:



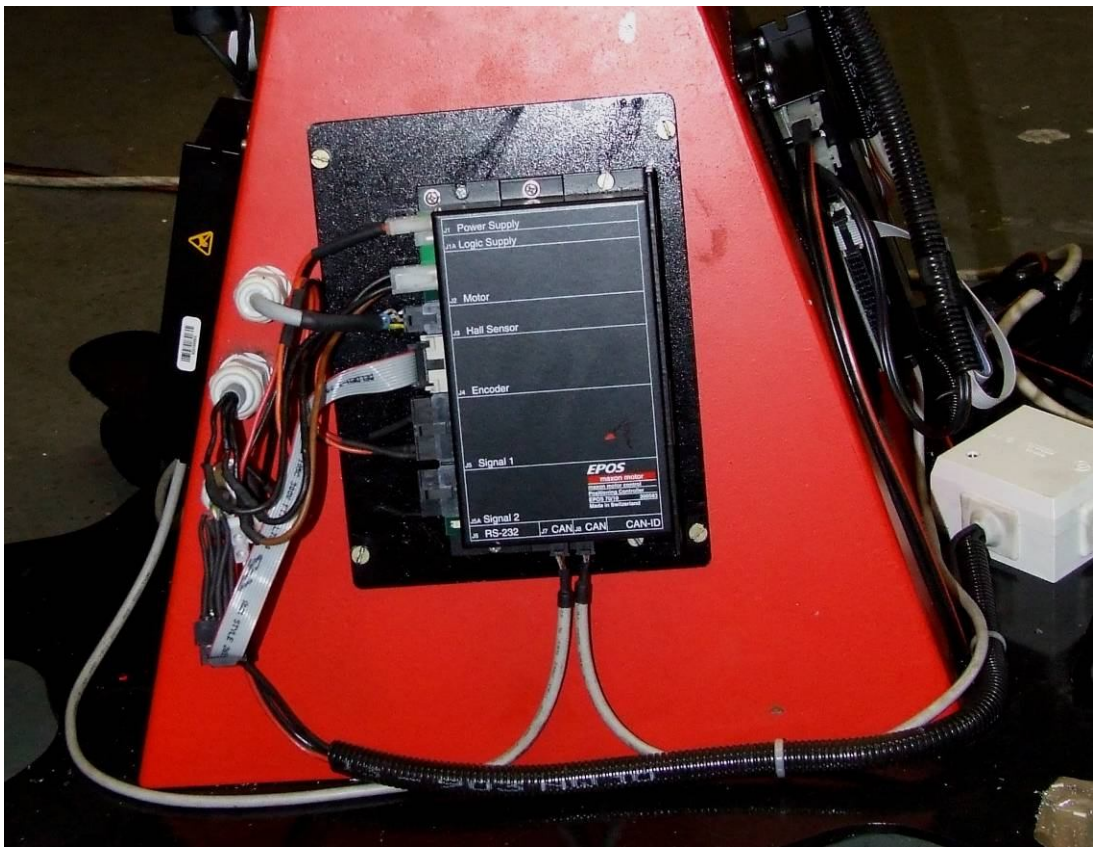
Obr.31 Celkový pohled na zdroje

Pro vedení výkonové části ve skříni se zdroji (58V / 21A) byly použity lanované kabelové vodiče 4x2,5 mm^2, kde vždy 2 vodiče jsou pro vedení kladného a 2 pro vedení záporného napětí od zdroje k výstupním svorkovnicím. Vodiče jsou dostatečně silné, takže nebude docházet k úbytku napětí a jejich zahřívání.

7. Elektroinstalace manipulátoru

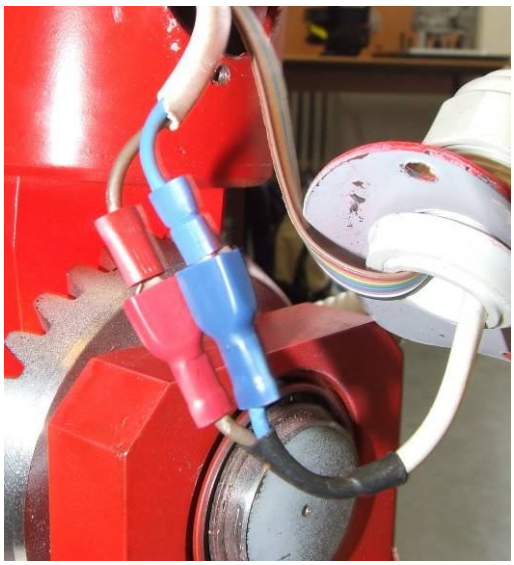
7.1 Připojení jednotlivých motorů

Připojení bezkartáčového motoru EC60 k jednotce EPOS #2 je provedeno přímo pomocí kabeláže, kterou je motor vybaven z výroby. Vodiče jsou protaženy průchodkami přes stojan robotu a jsou osazeny konektory ,které se zasunou přímo do jednotky EPOS. Pouze plochý kabel pro encodér bylo nutné prodloužit pomocí prodlužovacího propojky. (obr.32)

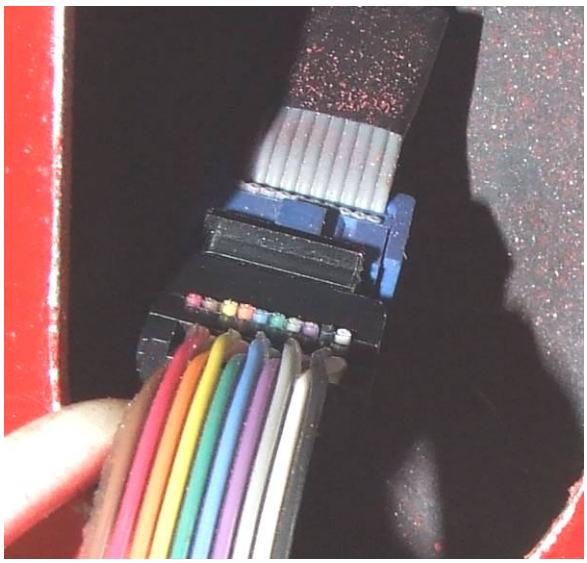


Obr.32 Pohled na zapojenou jednotku EPOS pro motor EC60

K ostatním motorům je vedena energie dvoužilovými kabely s průřezem 1 mm^2. Motory jsou připojeny pomocí konektorů fast-on (obr.33), aby bylo možné motory v případě potřeby snadno odpojit a případně vyměnit. Pro encodéry jsou použity ploché 10-ti žilové kabely s krimpovacími konektory. (obr.34)



Obr.33 Konektory fast-on motorů



Obr. 34 Spojení plochého kabelu encodéru

7.2 Nouzové tlačítko „Emergency STOP“

Robot je vybaven tlačítkem pro nouzové vypnutí v případě hrozícího nebezpečí nebo kolize. Tlačítko bude umístěno v dosahu operátora tak, aby by schopen v případě potřeby robota zastavit.

Obr.35 Tlačítko „Emergency STOP“

Popis funkce tlačítka:

Tlačítko obsahuje spínač se spínacími i rozpínacími kontakty, přičemž jsou použity pouze rozpínací kontakty. Ty jsou použity pro zabrzdění brzdy EC motoru (náklon ramene 1). Tlačítko přímo přeruší přívod energie pro napájení výstupů jednotky

EPOS a tím dojde k odpojení napájení brzdy. Tlačítko funguje při jakémkoli stavu jednotky (vypnutá / zapnutá, chyba), protože přerušuje přívod od zdroje (24V).

Dále je rozpínacích kontaktů tlačítka použito pro nastavení všech jednotek EPOS do pohotovostního režimu (DISABLED) a tedy i přerušení řízení pohonů. To se realizuje pomocí digitálního vstupu DI1 jednotek, který je nastaven jako ovládací vstup jednotek. Ztráta ovládacího signálu na vstupu jednotek způsobí jejich vypnutí. Jednotky tak okamžitě přestávají řídit motory. **Motory se však setrvačností ještě dotočí !! Pohony ramen nejsou samosvorné!! Pokud je rameno 2 zatíženo OM, může se jeho vahou sklopit voně dolů, protože pohon ramena 2 (osa 3) nemá brzdu!!**

8. Návrh algoritmu řídicího systému

8.1 Inverzní úloha kinematiky

Zatímco přímá úloha kinematiky při známé transformační matici je poměrně jednoduchá, inverzní transformace je velmi komplikovaná úloha. Je stále předmětem zkoumání mnoha robotických pracovišť. Její metody se stále vylepšují, stávají se univerzálnějšími a rychlejšími. Dnešní metody inverzní transformace jsou bez problémů použitelné u otevřených kinematických struktur do šesti stupňů volnosti jako v našem případě.

Inverzní úloha kinematiky přepočítává vektory polohy a orientace koncového bodu mechanismu na hodnoty kloubových proměnných. Je to základní úloha kinematiky a je nezbytná pro účely polohování jednotlivých článků mechanismu.

Vzhledem k tomu, že manipulátor není opatřen koncovým členem (efektor, nástroj,….) nebudeme se zabývat submaticí orientace koncového bodu, ale zaměříme se pouze na submatici polohy.

$$\mathbf{w} = \begin{bmatrix} \mathbf{p} \\ \mathbf{o} \end{bmatrix} \xrightarrow{inv.trans.} \mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \tag{3}$$

Pro použití inverzní úlohy kinematiky v řízení reálného zařízení, je důležitá rychlost výpočtu transformace, protože je třeba počítat inverzní transformaci v reálném čase. Vzhledem k velmi dynamickému chování robotů je třeba provádět inverzní transformaci cca každých 10 milisekund. Vzorkovací frekvence řídicího systému robotu by měla být tedy minimálně kolem 100 Hz.

Metody řešení inverzní úlohy

a) Vektorová metoda
b) Numerické metody
    i. Aproximační
    ii. Optimalizační
        1. Heuristické
        2. Gradientní

8.2 Hodnotová analýza

Hodnotovou analýzu použijeme pro hodnocení a následný výběr vhodné metody inverzní transformace pro řídicí systém manipulátoru.

Hodnotové analýze jsou podrobeny tyto varianty:

- aproximace Taylorovým rozvojem
- Newtonova aproximační metoda
- heuristická – CCD metoda
- gradientní metoda
- BFS metoda
- vektorová metoda
- metoda regulační smyčky

Popis metod inverzní transformace

1. Aproximace Taylorovým rozvojem

Homogenní transformační matice lze považovat za funkci více proměnných. Protože je tato funkce ve výchozím bodě diferencovatelná, lze tuto funkci v okolí výchozího bodu nahradit členy Taylorova rozvoje. Protože se jedná o iterační metodu, jsou použity pro výpočet pouze první dva členy Taylorova rozvoje.

Vlastnosti metody:

- citlivá na singulární polohu
- rychlá konvergence (asi 10 kroků)
- použitelná do 6 st. volnosti

2. Newtonova aproximační metoda

Podobně jako metoda Taylorovým rozvojem, využívá Newtonova metoda výpočtu inverzní transformace pomocí diferenciálu funkce ve výchozím bodě. Místo diferenciálu je tentokráte použito Jacobiovy matice. Její odvození je provedeno přímo pro danou kinematickou strukturu intuitivně.

Vlastnosti metody:

- citlivá na singulární polohu
- použitelná do 6 st. volnosti
- možnost řešení rychlostní úlohy
- potřeba přesného určení výchozí polohy

3. Heuristická – CCD

Pracuje pomocí fiktivního pohybování jednotlivých článků mechanismu tak, aby se minimalizovaly chyby polohy a orientace koncového souřadného systému. Začíná se fiktivním pohybem posledního článku tak, až je součet chyby polohování + chyby

orientace co nejmenší. Poté se pohybuje nižší jednotkou při fixaci všech ostatních a výpočet se opakuje.

Vlastnosti metody:

- velmi rychlá konvergence do bodu blízkého koncovému
- zpřesňování polohy je zdlouhavé
- necitlivá na singulární polohu
- použitelné i pro více st. volnosti
- pouze pro otevřenou kin. Strukturu
- není třeba přesného určení výchozí polohy

4. Newtonova gradientní metoda nalezení vektoru vyhledávání

Principem je tentokrát pohyb všemi pohybovými jednotkami současně tak, aby se hodnota chyby polohování pohybovala ve směru záporného gradientu (největšího spádu).

Vlastnosti metody:

- velmi dobrá konvergence
- výpočet Hessovy matice a vektoru vyhledávání je zdlouhavé
- není vhodná pro řízení v reálném čase

5. Gradientní metoda BFS

Principem je opět pohyb všemi pohybovými jednotkami současně tak, aby se hodnota chyby polohování pohybovala ve směru záporného gradientu (největšího spádu).

6. Vektorová metoda

Principem výpočtu u této metody je vyjádření kloubových proměnných mechanismu jako funkcí jeho geometrických vztahů mechanismu a polohy koncového bodu. Využívá

se goniometrických funkcí pro výpočet změn které nastanou při pohybu jednotlivých kloubů.

Vlastnosti metody:

- velmi jednoduchá
- snadno programovatelná
- citlivá na singulární polohu

7. Metoda regulační smyčky

Tato metoda je založena na výpočtu regulační odchylky. Inverzní úloha pracuje jako regulátor, přičemž se počítá rozdíl mezi žádanou a aktuální veličinou (poloha / rychlost) a nastavuje akční část dle velikosti tohoto rozdílu. Jedná se o experimentální metodu, která bude na manipulátoru testována pro ověření použitelnosti v řízení robotů.

Posouzení jednotlivých metod pomocí metody porovnávání v trojúhelníku párů

Všechny varianty řešení podrobíme hodnotové analýze. Pro tuto analýzu byla zvolena metoda porovnávání v trojúhelníku párů. Byla zvolena kritéria, podle kterých budeme varianty hodnotit a následně je rozebereme u každé z vybraných variant . Tyto varianty budou dále porovnány a pomocí kritéria významnosti vypočteme optimální variantu.

Hodnoty a významnost kritérií:

ÚROVEŇ	HODNOTA
vysoká	1
dobrá	2
průměrná	3
nízká	4
nevyhovující	5
nepříznivý stav	6

Tab.12 Hodnoty kritérií

VÝZNAMNOST	HODNOTA
nejvyšší	2
nejnižší	1

Tab.13 Významnost kritérií

Volba a hodnocení kritérií:

OZNAČENÍ KRITÉRIA	KRITÉRIUM	CHARAKTERISTIKA KRITÉRIA
K1	složitost výpočtu	složitost výpočtu inverzní transformace
K2	rychlost konvergence	doba do nalezení řešení
K3	citlivost na singulární polohu	schopnost algoritmu výpočtu inverze v singulární poloze
K4	snadnost programování	obtížnost realizace výpočtu v programovacím jazyku
K5	možnost řešení rychlostní úlohy	možnost použití rychlostního řízení

Tab. 14 Volba kritérií

Zhodnocení kritérií pro jednotlivé varianty:

KRIT.	varianta 1	varianta 2	varianta 3	varianta 4	varianta 5	varianta 6	varianta 7
K1	výpočet matic (velké množství operací)	výpočet matic (velké množství operací)	výpočet je složitý	zdlouhavý výpočet Hessovy matice	velmi složité pohyby – složité rovnice	jednoduché goniometrické vztahy	jednoduchý výpočet
	3	3	6	5	6	1	1
K2	rychlá (asi 10 iterací)	poměrně rychlá	rychlá do okolí řešení	poměrně rychlá	rychlá v blízkosti řešení	rychlá (v 1 iteraci)	rychlá, (během pohybu
	2	3	3	4	4	2	1
K3	citlivá	citlivá	necitlivá	necitlivá	necitlivá	citlivá	necitlivá
	6	6	1	1	1	6	1
K4	snadné	snadné	složité	velmi složité	velmi složité	velmi snadné	relativně snadné
	2	2	5	5	5	1	3
K5	možné	možné	možné	nelze v reálném čase	možné	značná nepřesnost	možné
	1	1	1	5	1	4	1

Tab. 15 Zhodnocení kritérií u jednotlivých variant

1. Odborník

POROVNÁVANÉ PÁRY KRITÉRIÍ				POČET VOLEB v	POŘADÍ
K1 K2	**K1** K3	K1 **K4**	**K1** **K5**	2,5	2-3
	K2 **K3**	**K2** **K4**	**K2** K5	1,5	4
		K3 **K4**	**K3** K5	2,5	2-3
			K4 K5	3	1
				0,5	5

Tab. 16 Dotazník významnosti a hodnocení odborníka 1

2. Odborník

POROVNÁVANÉ PÁRY KRITÉRIÍ				POČET VOLEB v	POŘADÍ
K1 K2	**K1** K3	**K1** K4	K1 **K5**	3	1-3
	K2 K3	K2 **K4**	K2 **K5**	1	4
		K3 **K4**	K3 **K5**	0	5
			K4 K5	3	1-3
				3	1-3

Tab. 17 Dotazník významnosti a hodnocení odborníka 2

3. Odborník

POROVNÁVANÉ PÁRY KRITÉRIÍ				POČET VOLEB v	POŘADÍ
K1 K2	K1 **K3**	K1 **K4**	**K1** K5	2	3-4
	K2 **K3**	K2 **K4**	K2 **K5**	0	5
		K3 **K4**	**K3** K5	3	1-2
			K4 **K5**	3	1-2
				2	3-4

Tab. 18 Dotazník významnosti a hodnocení odborníka 3

KOEFICIENT VÝZNAMNOSTI

KRITÉRIUM	KOEFICIENT VÝZNAMNOSTI q_i
K1	2,5
K2	0,833
K3	1,833
K4	3
K5	1,833

Tab. 19 Koeficient významnosti

Příklad výpočtu:

Koeficient významnosti q_i např. u kritéria K1, ostatní kritéria se počítají obdobným způsobem.

$$\frac{E1_1 + E2_1 + E3_1 + E4_1}{počet\ odborníků} = q_1 \qquad (4)$$

$$\frac{2,5+3+2}{3} = q_1$$

$$2,5 = q_1$$

$E1_1$ ..... počet voleb 1. experta u K1; $E2_1$ ..... počet voleb 2. experta u K1

$E3_1$ ..... počet voleb 3. experta u K1; $E4_1$ ..... počet voleb 4. experta u K1

Příklad výpočtu váženého indexu:

$$I^*_{ij} = I_{ij} * q_i \qquad (5)$$

$$I^*_{1A} = 1 * 3,250$$

$$I^*_{1A} = 3,250$$

q_i ..... váha významnosti i-tého parametru

I_{ij} ..... index změny i-tého parametru j-té varianty

VARIANTA A				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	3	2,5	3	7,5
K2	2	0,833	2	1,666
K3	6	1,833	6	11
K4	2	3	2	6
K5	1	1,833	1	1,833
Celkový součet indexů I^*_{ij} varianty A				**28**

Tab.20 Hodnocení varianty A

VARIANTA B				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	3	2,5	3	7,5
K2	3	0,833	3	2,5
K3	6	1,833	6	10,1
K4	2	3	2	6
K5	1	1,833	1	1,833
Celkový součet indexů I^*_{ij} varianty B				**27,933**

Tab.21 Hodnocení varianty B

VARIANTA C				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	6	2,5	6	15
K2	3	0,833	3	2,5
K3	1	1,833	1	1,833
K4	5	3	5	15
K5	1	1,833	1	1,833
Celkový součet indexů I^*_{ij} varianty C				**36,166**

Tab.22 Hodnocení varianty C

VARIANTA D				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	5	2,5	5	12,5
K2	4	0,833	4	3,333
K3	1	1,833	1	1,833
K4	5	3	5	15
K5	5	1,833	5	9,166
Celkový součet indexů I^*_{ij} varianty D				**41,831**

Tab.23 Hodnocení varianty D

VARIANTA E				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	6	2,5	6	15
K2	4	0,833	4	3,333
K3	1	1,833	1	1,833
K4	5	3	5	15
K5	1	1,833	1	1,833
Celkový součet indexů I^*_{ij} varianty E				**37**

Tab. 24 Hodnocení varianty E

VARIANTA F				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	1	2,5	1	2,5
K2	2	0,833	2	1,666
K3	6	1,833	6	11
K4	1	3	1	3
K5	4	1,833	4	7,333
Celkový součet indexů I^*_{ij} varianty F				**25,5**

Tab.25 Hodnocení varianty F

VARIANTA G				
KRITÉRIUM	**HODNOTA**	**VÁHA VÝZNAMNOSTI KRITÉRIÍ q_i**	**INDEX ZMĚNY I_{ij}**	**VÁŽENÝ INDEX KRITÉRIA I^*_{ij}**
K1	1	2,5	1	2,5
K2	1	0,833	1	0,833
K3	1	1,833	1	1,833
K4	3	3	3	9
K5	1	1,833	1	1,833
Celkový součet indexů I^*_{ij} varianty G				**16**

Tab. 26 Hodnocení varianty G

Vyhodnocení a určení nejvhodnější varianty

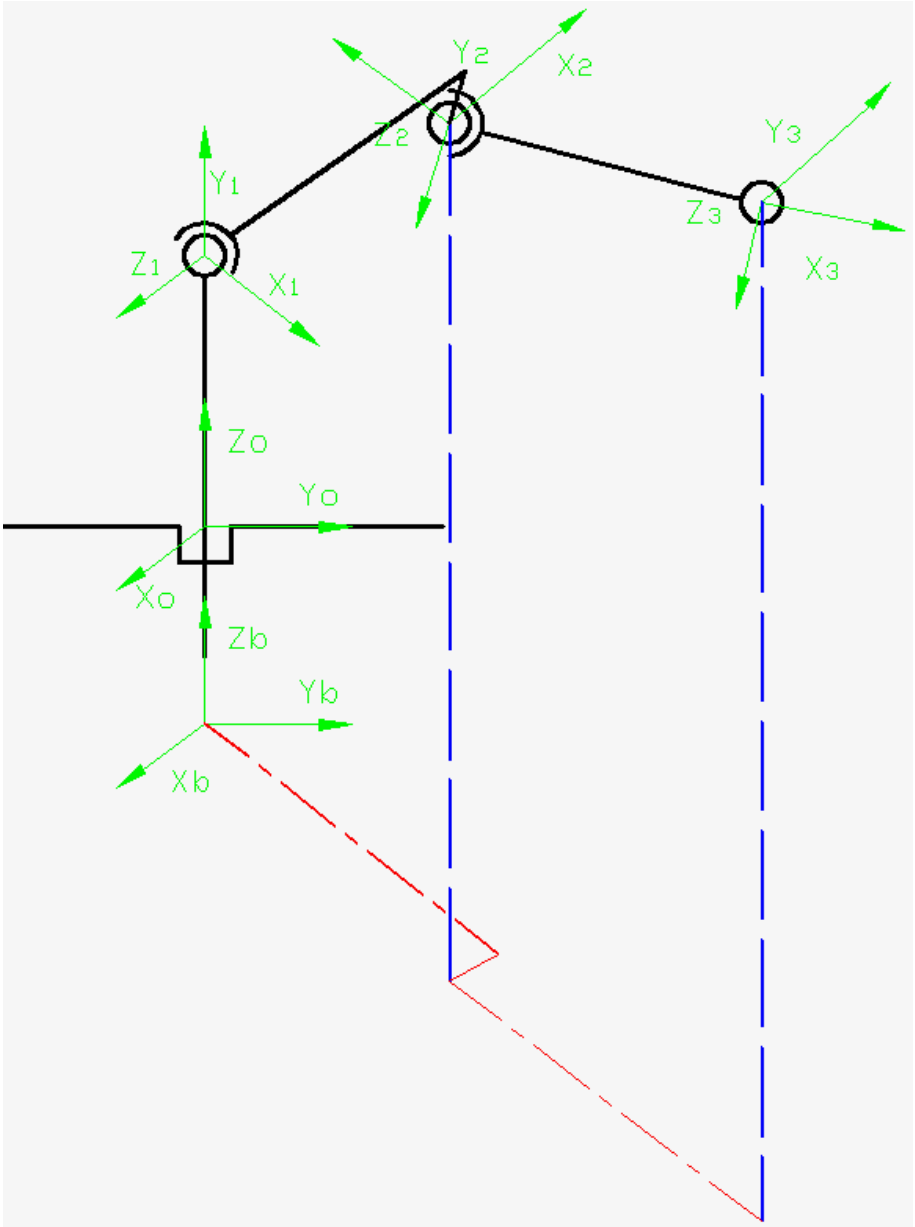
V následující tabulce je vidět, že nejlepší variantou je varianta G (metoda regulační smyčky), a jako druhá varianta F (vektorová metoda), které jsou popsány výše.

VARIANTA	CELKOVÝ SOUČET VÁŽENÝCH INDEXŮ I^*_{ij}	POŘADÍ
A	28	4.
B	27,933	3.
C	36,166	5.
D	41,831	7.
E	37	6.
F	25,5	2.
G	16	1.

Tab.27 Vyhodnocení analýzy

Vzhledem k tomu, že metoda regulační smyčky je pouze experimentální, použijeme pro výpočet inverzní úlohy metodu vektorovou, která vyšla jako druhá nejvýhodnější.

8.3 Kinematická struktura manipulátoru



Obr.36 Kinematické schéma manipulátoru

Délky ramen manipulátoru:

L_0 = 500 mm L_1 = 228,1 mm

L_2 = 600 mm L_{12} = 103 mm

L_3 = 400 mm

8.4 Transformační matice manipulátoru

Tabulka parametrů Denavit-Hatenberg:

Index	ϑ	d	a	α
1	0	L_0	0	0
2	Q_1	L_1	0	π/2
3	Q_2	L_{12}	L_2	0
4	Q_3	0	L_3	0

Tab. 28 Parametry D-H

Transformační matice mezi souřadnými systémy:

$$T_{B0} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6) \qquad T_{01} = \begin{pmatrix} \cos(q_1) & 0 & \sin(q_1) & 0 \\ \sin(q_1) & 1 & -\cos(q_1) & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (7)$$

$$T_{12} = \begin{pmatrix} \cos(q_2) & -\sin(q_2) & 0 & l_2\cos(q_2) \\ \sin(q_2) & \cos(q_2) & 0 & l_2\sin(q_2) \\ 0 & 0 & 1 & L_{12} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (8)$$

$$T_{23} = \begin{pmatrix} \cos(q_3) & -\sin(q_3) & 0 & L_3\cos(q_3) \\ \sin(q_3) & \cos(q_3) & 0 & L_3\sin(q_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (9)$$

Poslední sloupec celkové transformační matice T_{B3}, který vyjadřuje polohu koncového bodu mechanismu vůči bázovému souřadnému systému (podlaha) ve smyslu :

$$T_{B3}[4] = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{10}$$

$$T_{B3}[4] = \begin{pmatrix} \cos(q_1)\cos(q_2)L_3\cos(q_3) - \cos(q_1)\sin(q_2)L_3\sin(q_3) + \cos(q_1)L_2\cos(q_2) + \sin(q_1) * L_{12} \\ \sin(q_1)\cos(q_2)L_3\cos(q_3) - \sin(q_1)\sin(q_2)L_3\sin(q_3) + \sin(q_1)L_2\cos(q_2) - \cos(q_1) * L_{12} \\ \sin(q_2)L_3\cos(q_3) + \cos(q_2)L_3\sin(q_3) + L_2\sin(q_2) + L_1 + L_0 \\ 1 \end{pmatrix} \tag{11}$$

Výpočet polohy pomocí transformační matice mezi bázovým a koncovým souřadným systémem (přímá úloha):

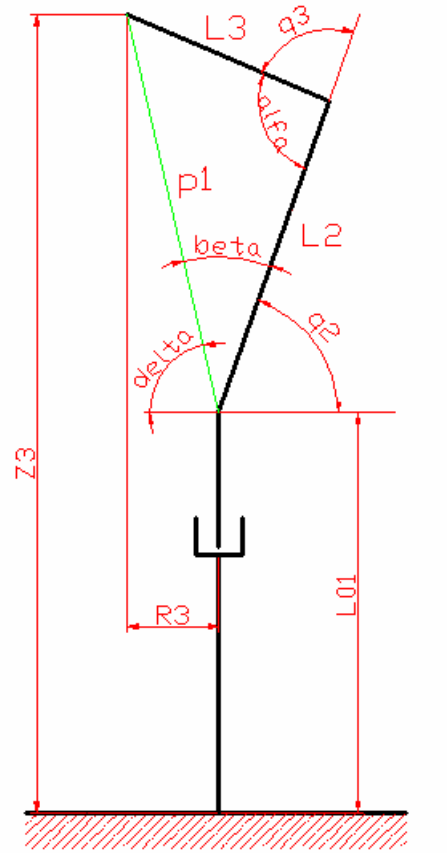
X = cos (q1) * cos (q2) * L3 * cos (q3) - cos (q1) * sin (q2) * L3 * sin (q3) + + cos (q1) * cos (q2) * L2 + sin (q1)* L12 (12)

Y = sin (q1) * cos (q2) * L3 * cos (q3) - sin (q1) * sin (q2) * L3 * sin (q3) + + sin (q1) * cos (q2) * L2 + cos (q1)* L12 (13)

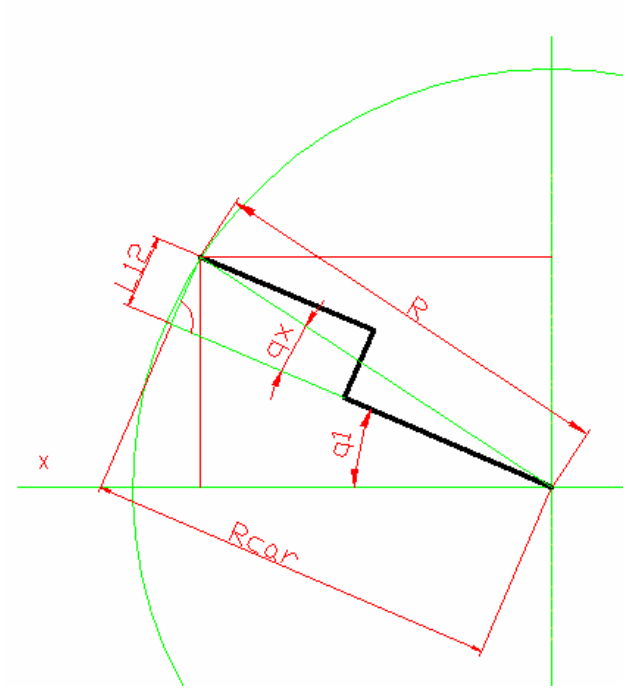
Z = sin (q2) * L3 * cos (q3) + cos (q2) * L3 * cos(q3) + L2 * sin(q2) + L0 + L1 (14)

8.5 Inverzní úloha pomocí vektorové metody

Schéma pro výpočet inverzní úlohy vektorovou metodou:



Obr.37 Schéma pro výpočet inverzní úlohy (nárys)



Obr.38 Schéma pro výpočet inverzní úlohy (půdorys)

Výpočet inverzní úlohy pomocí vektorové metody:

$$R_3 = \sqrt{x_3^{\,2} + y_3^{\,2}} \quad (15) \qquad R_3^{\,2} = x_3^{\,2} + y_3^{\,2} \quad (16)$$

$$R_{3CAR} = \sqrt{R_3^{\,2} - L_{12}^{\,2}} \quad (17)$$

$$P_1 = \sqrt{R_{3CAR}^{\,2} + (Z_3 - L_{10})^2} \quad (18) \qquad p = \sqrt{R_{3CAR}^{\,2} + (Z_3 - L_{10})^2 + L_{12}^{\,2}} \quad (19)$$

$$P_1 = \sqrt{L_3^{\,2} + L_2^{\,2} - 2L_3L_2\cos\alpha} \quad (20)$$

$$p = \sqrt{\left(L_3^{\,2} + L_2^{\,2} - 2L_3L_2\cos\alpha\right) + L_{12}^{\,2}} \quad (21)$$

$$P_1 = P_1 \quad (22)$$

$$R_{3CAR}^{\,2} + (Z_3 - L_{10})^2 = L_3^{\,2} + L_2^{\,2} - 2L_3L_2\cos\alpha \quad (23)$$

$$\frac{R_{3CAR}^{\,2} + (Z_3 - L_{10})^2 - L_3^{\,2} - L_2^{\,2}}{2L_3L_2} = -\cos\alpha \quad (24)$$

$$\alpha = \arccos\left(-\frac{R_{3CAR}^{\,2} + (Z_3 - L_{10})^2 - L_3^{\,2} - L_2^{\,2}}{2L_3L_2}\right) \quad (25)$$

$$\underline{q_3 = \pi - \alpha} \quad (26)$$

$$\tan\delta = \frac{(Z_3 - L_{10})}{R_{3CAR}} \quad (27) \qquad \delta = \arctan\frac{(Z_3 - L_{10})}{R_{3CAR}} \quad (28)$$

$$L_3^{\,2} = P_1^2 + L_2^{\,2} - 2P_1L_2\cos\beta \quad (29) \qquad \cos\beta = \frac{L_2^{\,2} - L_3^{\,2} + P_1^2}{2P_1L_2} \quad (30)$$

$$\beta = a\cos\left(\frac{L_2^{\,2} - L_3^{\,2} + P_1^2}{2P_1L_2}\right) \quad (31)$$

$$\underline{q_2 = \pi - \beta - \delta} \quad (32)$$

$$\cos q_X = \frac{R_{CAR}}{R} \quad (33) \qquad q_X = a\cos\left(\frac{R_{3CAR}}{R_3}\right) \quad (34)$$

$$\underline{q_1 = a\tan 2(-X, -Y) - qx} \quad (35)$$

Atan2 je funkce, která počítá arkus tangens podílu parametrů správně ve všech čtyřech kvadrantech. Funkce je dostupná ve většině programovacích jazyků včetně jazyka BASIC.

Meze pracovního prostoru:

Vzhledem ke své kinematické struktuře není robot schopen pokrýt pohybem pracovní prostor ve tvaru válce s průměrem 2xL12 a osou rotace kolem osy z1.

Pro q1 = 0° **xmin=± L12**

Pro q1 = 90° **ymin=± L12**

Vzhledem ke konstrukci manipulátoru je omezen jeho dosah od osy rotace z důvodu maximálního náklonu ramena 1 od svislé osy 73°. Proto je ve výpočtu inverzní úlohy řídicího programu provedena korekce q2 o 17° od vodorovné roviny.

8.6 Singulární poloha a problémy výpočtu

Slabé místo výpočtu inverzní transformace pomocí vektorové metody je v singulární poloze mechanismu. V tomto případě není algoritmus schopen výpočet provést. Z toho důvodu je nutné v okolí singulární polohy provést interpolaci kloubových proměnných z předchozích hodnot.

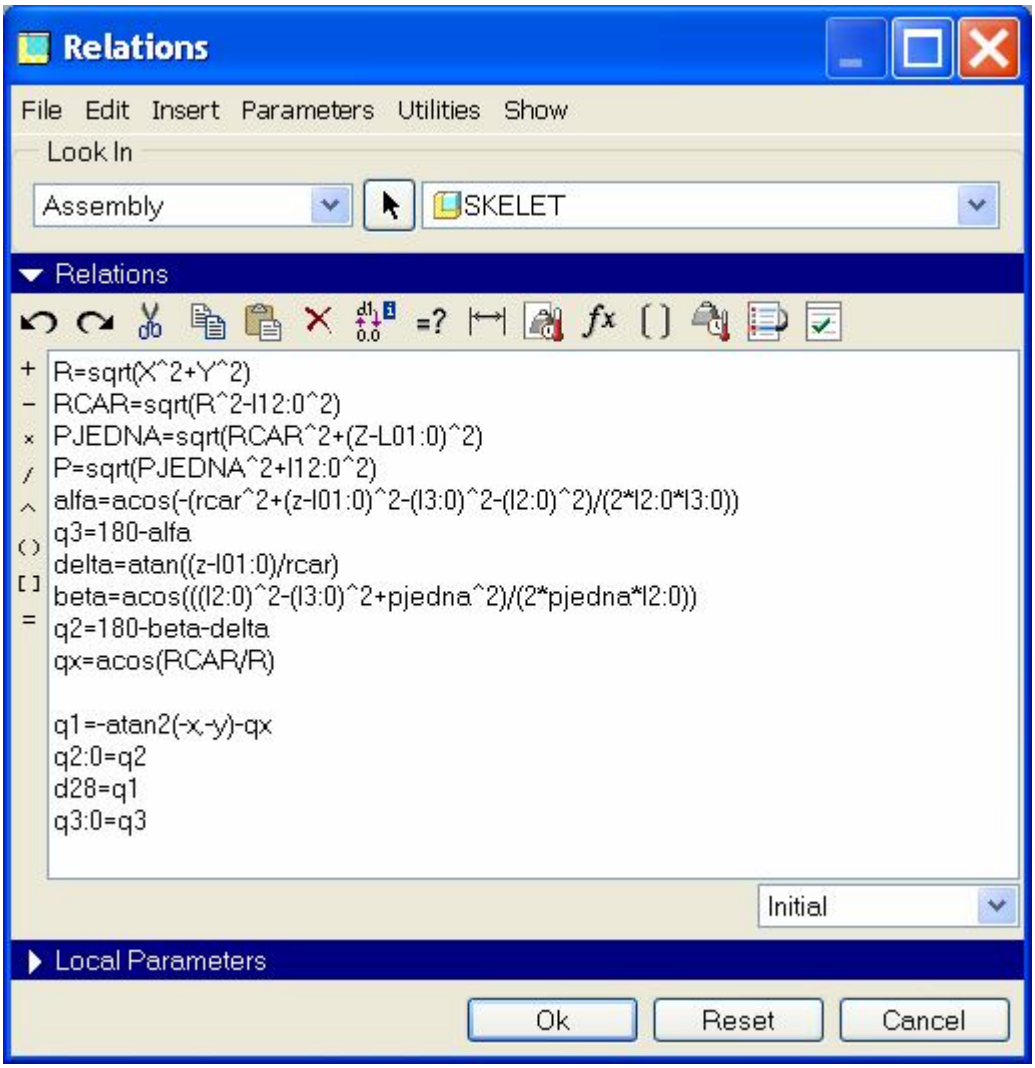
Dalším problémem při výpočtu inverzní úlohy je neznalost přesného skutečného náklonu ramen při inicializaci. Počáteční úhly, ve kterých jsou kloubové proměnné rovny 0 jsou jen přibližně odhadnuty. Tato skutečnost vnáší do výpočtu jistou nepřesnost a tím i nepřesnost polohování manipulátoru. Tato nepřesnost je nejvíce zřetelná při přímkové interpolaci (koncový bod nesleduje přímku ale křivku).

Odstranění této nepřesnosti by částečně pomohlo osazení ramen manipulátoru inklinometrem nebo akcelerometrem s analogovým výstupem. Ten by bylo možné připojit na analogové vstupy jednotek EPOS. Došlo by tak k usnadnění počáteční inicializace jednotlivých os, a navíc by bylo možno kompenzovat vůle v převodech.

8.7 Simulace a kontrolní výpočty

Simulace inverzní úlohy pomocí systému Pro/Engineer

Provedl jsem simulaci výpočtu inverzní úlohy vektorovou metodou pomocí Skeleton modelu v systému Pro/Engineer. Byl zhotoven hrubý model navázaný na skeleton, jehož geometrie (poloha) byla vypočítávána dle zadaných souřadnic koncového bodu robotu. Následně byla provedena kontrola správnosti polohování pomocí Measure - Transform, která ukazuje transformační matice mezi souřadnými systémy. Kontrola potvrdila správnost výpočtu inverzní transformace vektorovou metodou.



Obr.39 Okno relací v Pro/Engineeru pro výpočet inverzní úlohy

Obr.40 Výpočtový model skeletonu

Výpočet inverzní úlohy pomocí systému MathCAD 13

Provedl jsem simulaci výpočtu inverzní úlohy vektorovou metodou pomocí programu MathCAD 13. Pro následnou kontrolu správnosti výpočtu jsem provedl výpočet polohy koncového bodu pomocí transformační matice mezi bázovým a koncovým souřadným systémem. Tento výpočet také potvrdil správnost vektorové inverzní transformace.

8.8 Konfigurace jednotek před použitím

Jednotky EPOS je třeba před použitím nakonfigurovat. Jsou zapotřebí katalogové listy příslušného motoru a encodéru připojeného k jednotce EPOS. Ke konfiguraci je použit dodávaný software „EPOS User Interface“, určený pro konfiguraci a kalibraci jednotek. Do jednotek je nutno zadat:

- typ motoru a počet pólů
- maximální otáčky a proud motoru
- typ snímače polohy pohonu (encodér, tachodynamo, hallovy sondy)
- vstupy a výstupy (brzda, koncové spínače, atd.….)

Dalším krokem nastavení jednotek je kalibrace zesílení jednotlivých regulátorů a jejich složek v jednotkách. Tato kalibrace se provádí také v dodávaném softwaru „EPOS User Interface“, případně „EPOS Studio“.

Kalibrace se provede pro proudový, rychlostní a polohový regulátor. Program si sám vypočte a nastaví potřebné parametry jednotlivých složek PI regulátoru (proudový, rychlostní) a PID regulátoru (polohový).

Konfigurace jednotek EPOS:

Adresa Jednotky	Funkce	Druh motoru	Typ motoru	Výkon Motoru
Node 1	rotace základu	DC	RE36	70 W
Node 2	náklon ramena 1	EC	EC60 + brzda	400 W
Node 3	náklon ramena 2	DC	F2260	80 W
Node 4	rotace příruby	DC	RE36	70 W

Tab. 29 Přiřazení jednotek EPOS pohonům

Protože pohony neobsahují absolutní snímače polohy, je nutné tuto funkci zajistit softwarově. Tuto funkci plní řídicí jednotka, která je schopna si pamatovat absolutní polohu pohonu od doby, kdy je zapojeno její napájení. Po provedení

synchronizace s koncovým spínačem nebo home spínačem, jsou jednotky schopny počítat s absolutní polohou pohonu do hodnoty ±2 147 483 648 qc.

Limit softwarového čítače encodéru: -2 147 483 648 / +2 147 483 648

9. Software řídicího systému pro řízení robotu

9.1 Vývojové prostředí

Pro řízení robotu je použit programovací jazyk Basic za pomocí Visual Studia 2008. Jsou použity dvě komponenty dodávané firmou MAXON v podobě dynamické knihovny „eposcmd.dll“ a definičního souboru „definitions.bas“.

Soubor definitons.bas je určen pro Visual Basic 6.0, proto bylo třeba jej zkonvertovat na platný formát pro Visual Studio 2008. Konverzi provedl průvodce pro konverzi projektů ve Visual studiu 2008. Výsledný soubor je „definitions.vb“ a obsahuje odkazy na funkce, které jsou definované v knihovně „eposcmd.dll“.

Dále byly použity části zdrojového kódu ze vzorového programu určeného pro Visual Basic 6.0, který je také součástí podpůrné dokumentace dodávané k jednotkám EPOS.

Všechny potřebné použité funkce pro řízení a komunikaci s jednotkami EPOS byly převzaty z dokumentu „EPOS Windows 32-Bit DLL.pdf“

9.2 Vlastnosti a možnosti softwaru

Vzhledem k absenci absolutních snímačů polohy na pohonech je třeba robot po zapnutí synchronizovat. Řídicí software je vybaven automatickou synchronizací všech os pomocí nájezdu na jednotlivé koncové spínače každé osy. Tím se určí jednoznačně poloha jednotlivých os. Od této chvíle tato poloha zůstává v jednotkách EPOS uložena, dokud nejsou odpojeny od napájení. V případě potřeby či ztráty napájení je nutné provést synchronizaci znovu.

Princip funkce synchronizace manipulátoru

Pro synchronizaci manipulátoru je použito režimu „6 - Homing mode“ jednotek EPOS. Je použito varianty pro synchronizaci s koncovými spínači „Positive limit“ a zpřesnění (odstranění hystereze spínače) pomocí signálu „Index“ z encodéru (submód 2). Je možné provést synchronizaci všech os najednou nebo každé osy zvlášť.

9.3 Postup uvedení manipulátoru do chodu

Postup při zapnutí robotu

Před použitím manipulátoru je nutné připojit jej ke zdroji napájení. Provede se připojení konektoru pro napájení brzdy a koncových spínačů (24V). Dále se připojí silová část na šroubové svorky (černé kabely na + 58V , hnědý a žlutozelený na -58V). Poté je možné zapnutím síťového vypínače na skříni se zdroji uvést manipulátor do pohotovostního režimu.

Následně se provede spuštění aplikace pro řízení manipulátoru.

Po spuštění se stiskem tlačítka „Inicializace“ naváže komunikace s manipulátorem. Od této chvíle aplikace kontroluje stavy všech jednotek a motorů a také načítá polohu motorů. **Jednotky jsou však stále v pohotovostním režimu a vypnuté!! Brzda zůstává dále zabrzděná.**

Pokud byla inicializace úspěšná, program se ptá, jestli byl manipulátor synchronizován (po připojení napájení). Pokud ano, můžeme tento dotaz stornovat. Pokud ne, stiskem tlačítka „OK“ provede program synchronizaci jednotlivých os manipulátoru.

Po provedení synchronizace je již manipulátor připraven pro použití.

Stiskem tlačítka „Zapnout řídicí systém“ se jednotky EPOS zapnou a motory začnou držet svou současnou polohu. Současně se také připojí napájení k brzdě a **brzda přestává brzdit!!**

9.4 Řízení manipulátoru

Software je vybaven ručním i automatickým chodem robotu. Je možné ho řídit několika způsoby:

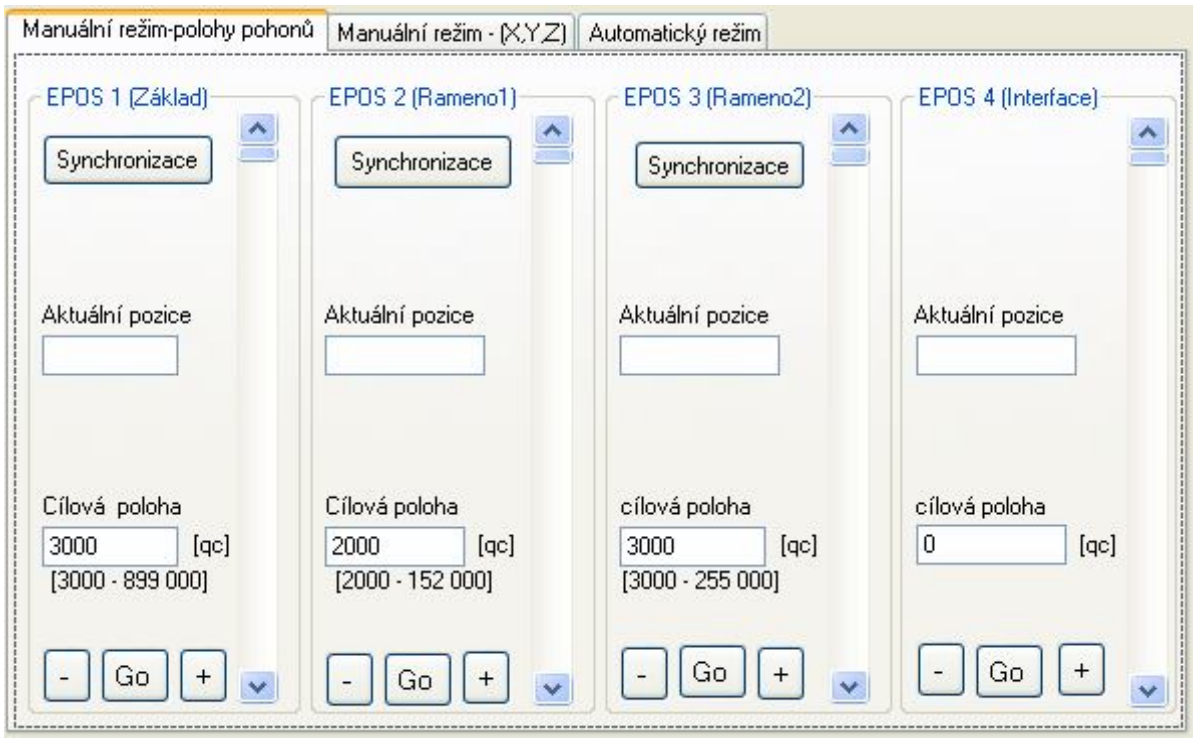
- ruční zadávání polohy pohonu (qc)
- zadávání polohy koncového bodu (X,Y,Z) [mm]
- načtením uloženého programu z paměti PC

V automatickém režimu je možný provoz robotu:

- Pohyb koncového bodu po nedefinované trajektorii na zadané souřadnice (řízení Point – to - Point)
- Pohyb koncového bodu po přímce na zadané souřadnice (lineární interpolace)

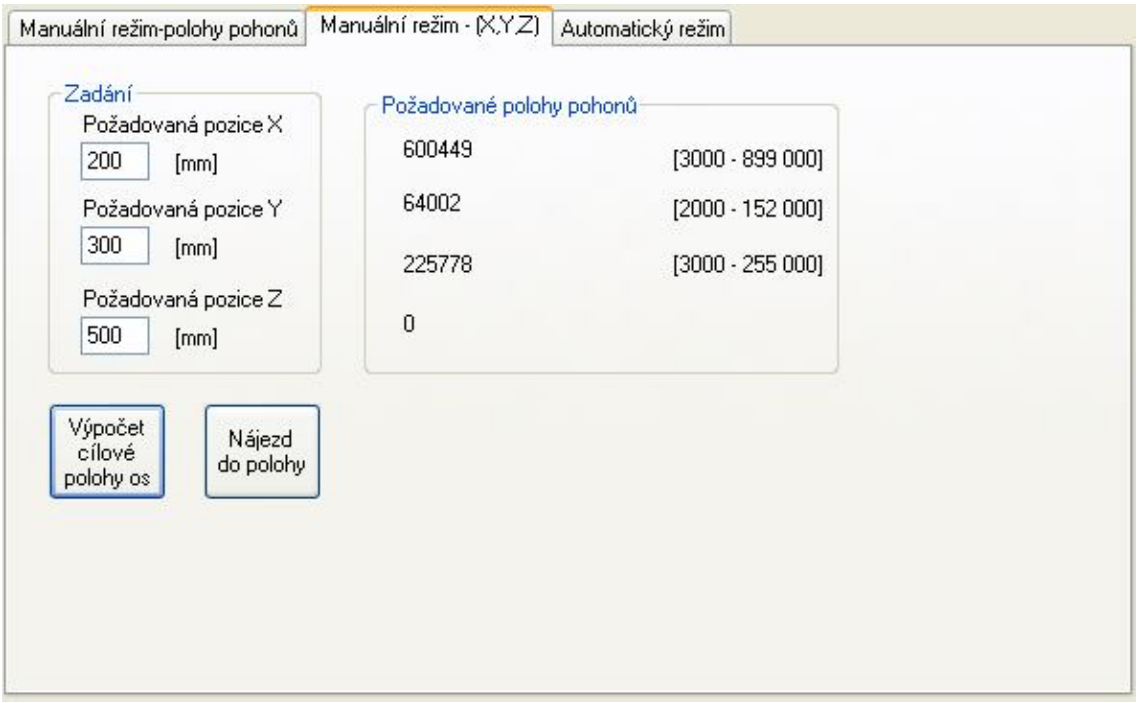
Ovládání jednotlivých pohonů v okně „Manuální režim – polohy pohonů“

- vertikálním posuvníkem (rozsah = rozsah osy)
- zadáním požadované polohy v [qc] a stiskem „GO“
- stiskem tlačítka „+“ nebo „ - “ se pohon pootočí o 5000 qc (1 otáčka motoru = 2000 qc)



Obr.41 Okno manuálního řízení pohonů

Zadávání polohy koncového bodu v kartézských souřadnicích v okně „Manuální režim – (X,Y,Z)“

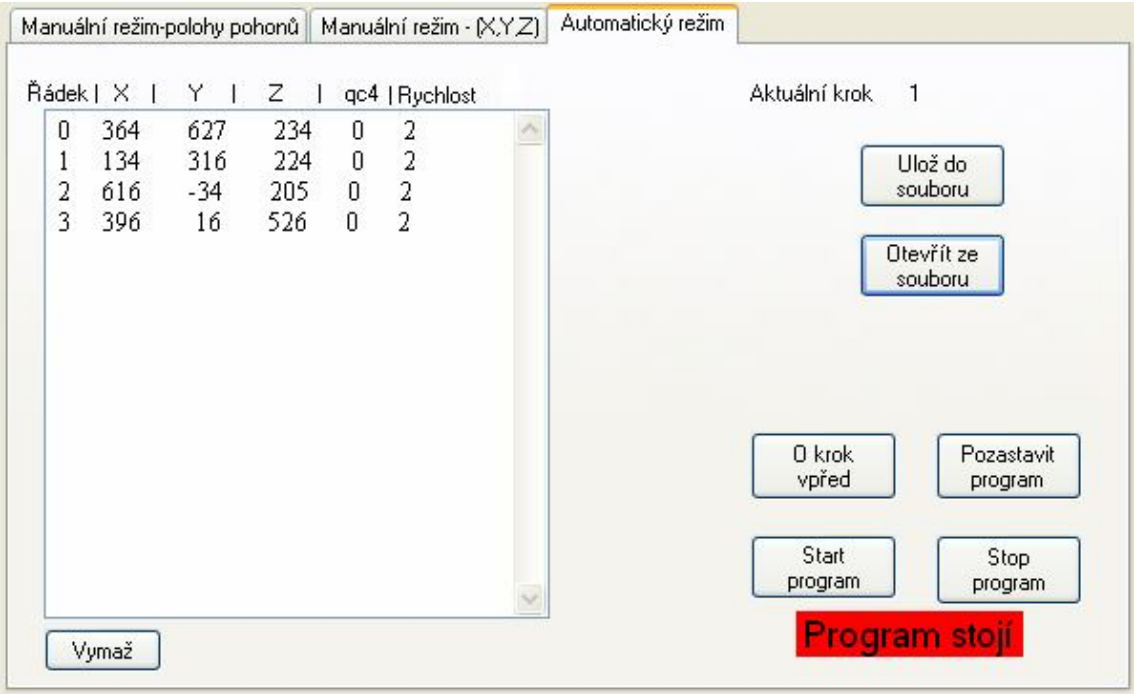


Obr.42 Okno manuálního řízení koncového bodu X,Y,Z

Robot se programuje standardním principem učení postupem:

1. Nájezd na požadované souřadnice v ručním režimu
2. Uložení aktuálních souřadnic do paměti
3. Nájezd na další souřadnice
4. Uložení aktuálních souřadnic do paměti
5. Uložení celého programu do souboru

Uložený program je možné také podle potřeby krokovat, či pozastavit.



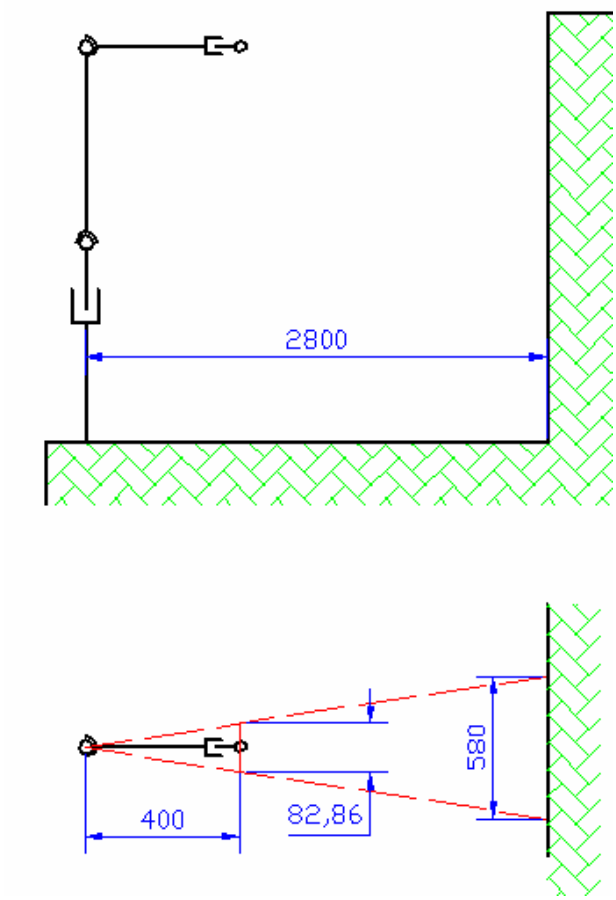
Obr.43 Okno programovacího režimu

9.5 Testování a experimentální měření

9.6 Měření vůlí jednotlivých os

Na manipulátoru bylo provedeno měření vůlí každé osy. Toto měření bylo provedeno pomocí laserového ukazovátka připevněného na přírubu. Motory byly aktivovány aby držely svou polohu. Rameny bylo pohybováno ručně a v krajních polohách vymezujících vůli byl zaznamenán bod dopadu paprsku laserového ukazovátka na stěnu. Při známé vzdálenosti stěny a délkách ramen bylo možné vypočíst vůli na koncovém bodu manipulátoru způsobenou danou osou. V tabulkách jsou uvedeny naměřené hodnoty, které jsou již přepočteny na koncový bod manipulátoru. Výpočet byl proveden dle obrázku za použití vztahů pro podobnost trojúhelníků.

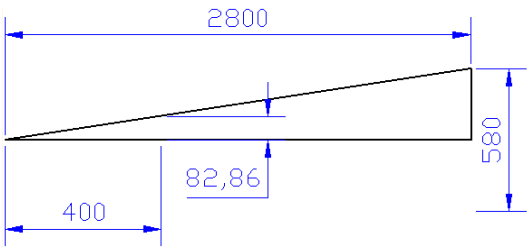
Měření vůlí v ose 1 (rotace základu kolem svislé osy)



Obr.44 Schéma měření vůle rotace základu

Měření č.	Vůle v koncovém bodě [mm]
1	0
2	- 40,5
3	+ 41,06
4	- 39,8
5	+ 41,2
6	- 40,8
7	+ 39,4
8	- 41,8
9	+ 41,1
10	- 40,9
Průměrná odchylka	± **40,94**
Maximální odchylka	**- 41,8** **+ 41,06**

Tab.30 Naměřené hodnoty osy 1



Obr.45 Schéma pro výpočet

Příklad výpočtu:

2800 mm …….. 580 mm

400 mm ……. X mm

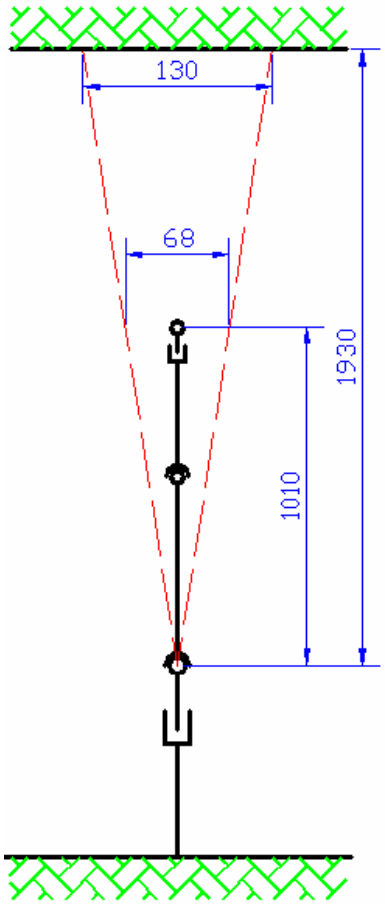
$$\frac{580*400}{2800} = 82,86mm \qquad (36)$$

Z měření vyplývá, že vůle osy 1 se pohybuje v rozmezí -41,8 mm až +41,06 mm. Celková vůle je tedy 82,86 mm. Tato vůle je způsobena výrobními nepřesnostmi spojovacích součást, hřídelů a také zubovou vůlí v převodovce motoru z důvodu vysokého převodového poměru.

Při maximálním vyložení obou ramen (983 mm od osy rotace) by tato vůle činila 205 mm, což je naprosto nevyhovující.

Navíc došlo během používání manipulátoru k uvolnění motoru osy 1 z rámu, takže se projevila navíc vůle v tomto uložení. Doporučují proto opravu této osy.

Měření vůlí v ose 2 (náklon ramena 1)



Obr.46 Schéma měření vůle ramena 1 (osa 2)

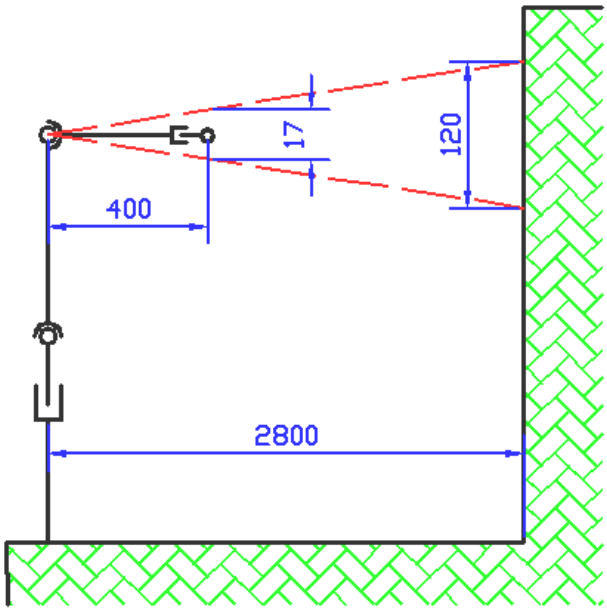
Měření č.	Vůle v koncovém bodě [mm]
1	**0**
2	- 33
3	**+ 32**
4	- 34
5	**+ 33**
6	- 34
7	**+ 32**
8	- 32
9	**+ 34**
10	- 33
Průměrná odchylka	± **33**
Maximální odchylka	**- 34** **+ 34**

Tab.31 Naměřené hodnoty osy 2

Z uvedeného měření vyplývá, že vůle osy 2 se pohybuje v rozmezí -34 až +34 mm. Celková vůle tedy bude 68 mm. Tato vůle je způsobena výrobními a montážními

nepřesnostmi nábojů, hřídelů a také zubovou vůlí v převodovce motoru. Kuželové soukolí osy je v tomto případě bez vůle.

Měření vůlí v ose 3 (náklon ramena 2)



Obr.47 Schéma měření vůle ramena 2 (osa 3)

Měření č.	Vůle v koncovém bodě [mm]
1	0
2	+7
3	-8
4	+6
5	-6
6	+8
7	-7
8	+9
9	-7
10	+8
Průměrná odchylka	**± 7,5**
Maximální odchylka	**- 8** **+ 9**

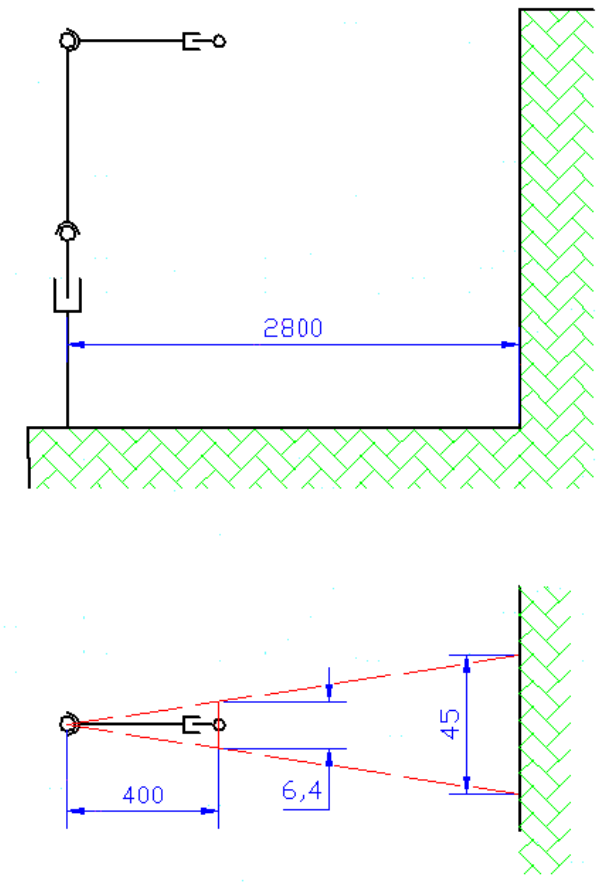
Tab.32 Naměřené hodnoty osy 3

Z uvedeného měření vyplývá, že vůle osy 3 se pohybuje v rozmezí -8 až +9 mm. Celková vůle tedy bude 17 mm. Tato vůle je způsobena výrobními nepřesnostmi spojovacích součástí hřídelů, zubovou vůlí v kuželovém ozubení pohonu osy a také zubovou vůlí v převodovce motoru.

9.7 Měření opakovatelné přesnosti polohování

Manipulátor byl podroben experimentálnímu měření opakovatelné přesnosti. Postupovalo se obdobným způsobem jako při předchozích měřeních. Při měření bylo opět použito laserové ukazovátko. Opět se zaznamenal bod dopadu paprsku na stěnu. Manipulátor poté provedl pohyby v měřené ose a následně se vrátil do výchozí pozice. Paprsek ukázal na stěně další bod. Měření se opakovalo desetkrát. Naměřené odchylky, přepočtené na koncový bod, jsou uvedeny v tabulce. Při známé vzdálenosti stěny bylo možné vypočíst opakovatelnou přesnost polohy příruby manipulátoru. Měření byla prováděna pro každou osu zvlášť, abychom zjistili jakou složkou přispívá každá osa k celkové nepřesnosti manipulátoru.

Měření opakovatelné přesnosti v ose 1 (rotace základu kolem svislé osy)



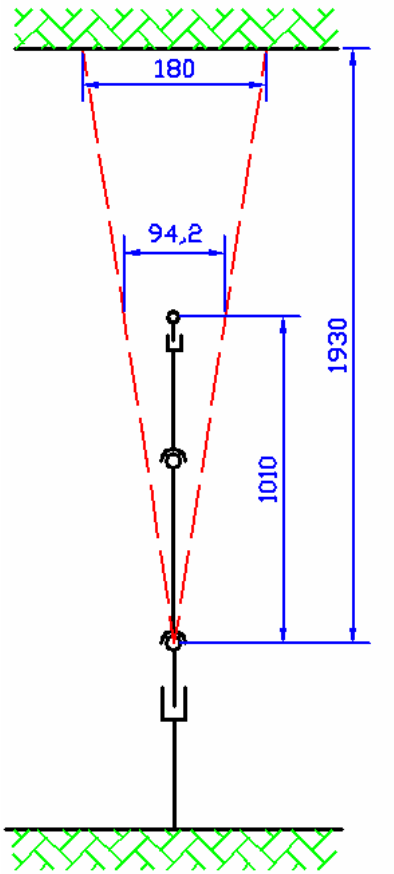
Obr.48 Schéma měření přesnosti na ose 1

Č. měření	Odchylka [mm]
1	0
2	+ 3
3	- 2
4	+ 3,2
5	- 2
6	+ 3
7	- 3,2
8	+ 1
9	- 2,5
10	+ 3
Průměrná odchylka	**± 2,54**
Maximální odchylka	**- 3,2** **+ 3,2**

Tab.33 Naměřené hodnoty osy 1

Opakovatelná přesnost osy 1 se pohybuje v přijatelných mezích. Neprojevují se zde vůle v převodech. Osa 1 je schopna opakovatelné přesnosti 6,4 mm. Torzní tuhost osy je však kvůli řemenovému převodu značně velká (projevila se v měření vůle). Manipulátor tak není možno použít pro nesení technologického nástroje. Síly od technologického procesu by způsobily natažení řemenu a také by se projevila velká vůle osy.

Měření opakovatelné přesnosti v ose 2 (náklon ramena 1)



Č. měření	Odchylka [mm]
1	**0**
2	+ 30
3	- 46
4	+ 45
5	- 40
6	+ 29
7	- 34
8	+ 48,2
9	- 43
10	+ 24
Průměrná odchylka	± 37,7
Maximální odchylka	- 46 + 48,2

Obr.49 Schéma měření přesnosti na ose 2

Tab.34 Naměřené hodnoty osy 2

V měření opakovatelné přesnosti manipulátoru v této poloze, se projeví v měření již dříve naměřené vůle v převodech. Pokud tyto vůle odečteme od naměřené přesnosti získáme přibližnou hodnotu opakovatelné přesnosti.

Opakovatelná přesnost = odchylka – vůle = 94,2 – 68 = 28,2 mm (36)

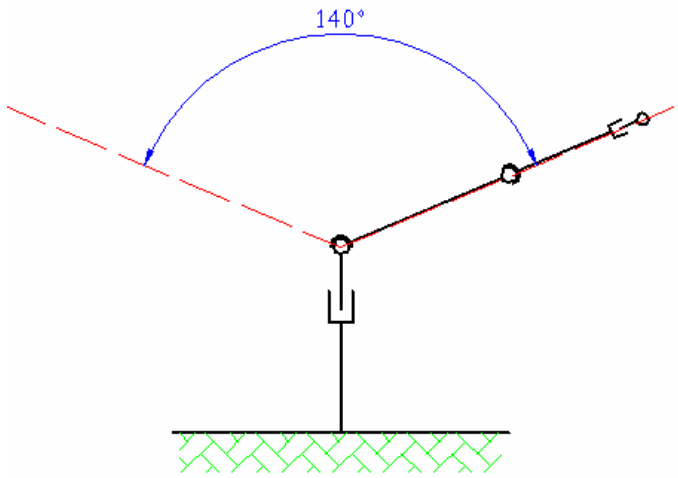
Po odečtení vůlí se přesnost pohybuje kolem 28 mm v koncovém bodě.

Měření opakovatelné přesnosti bylo provedeno i pro osu 3 (náklon ramena 2). Rameno bylo měřeno ve vodorovné poloze. Při tomto měření však rameno vykazovalo téměř nulovou odchylku. Hodnota opakovatelné přesnosti se pohybovala v neměřitelném rozsahu (desetiny milimetru).

9.8 Měření proudu (odběru) pohonů

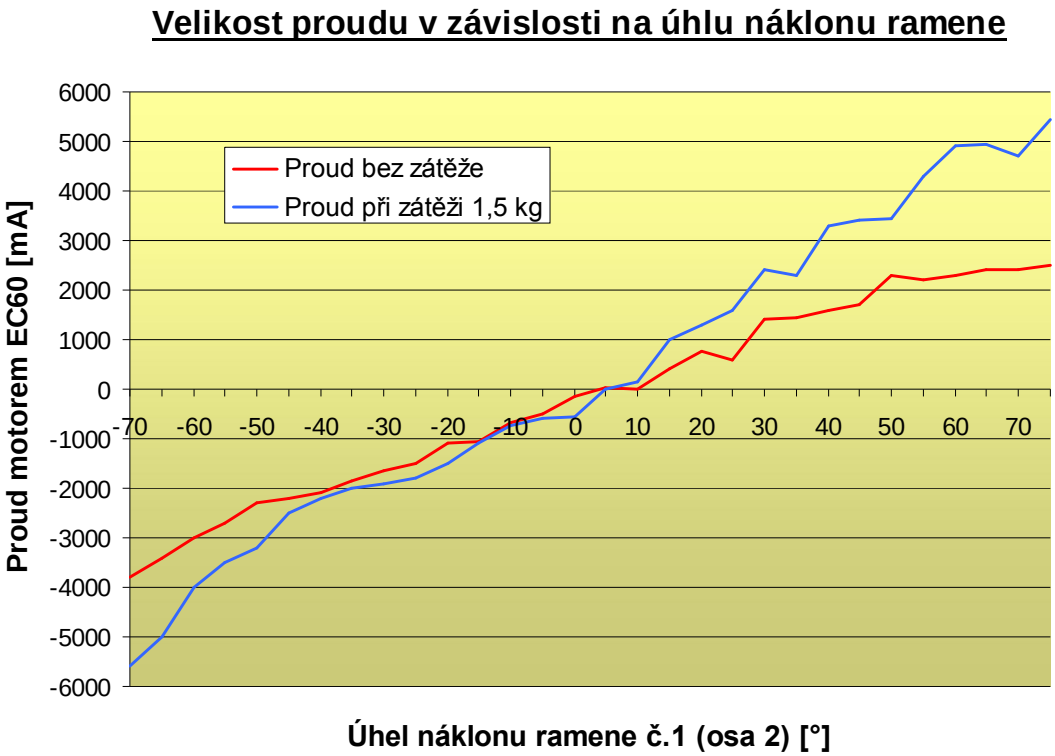
Cílem tohoto měření bylo získat orientační informaci o průběhu momentu v osách. Pro zjištění velikosti proudů odebíraných jednotlivými motory byla použita funkce jednotky EPOS pro zjištění aktuálního proudu motorem (VCS_Getcurrentis). Funkce je součásti knihovny „eposcmd.dll“ a její použití je velmi snadné.

Měření bylo provedeno pouze pro osu 2, protože osa 1 svůj motor při ustáleném stavu nezatěžuje. Osa 3 používá motor s permanentními magnety, který vyvíjí moment dostatečný pro udržení ramena 2, i bez napájení.



Obr.50 Schéma měření proudu motoru EC60

Naměřené proudy motoru EC60 při různém úhlu náklonu ramena 1 při maximálním vyložení ramena 2



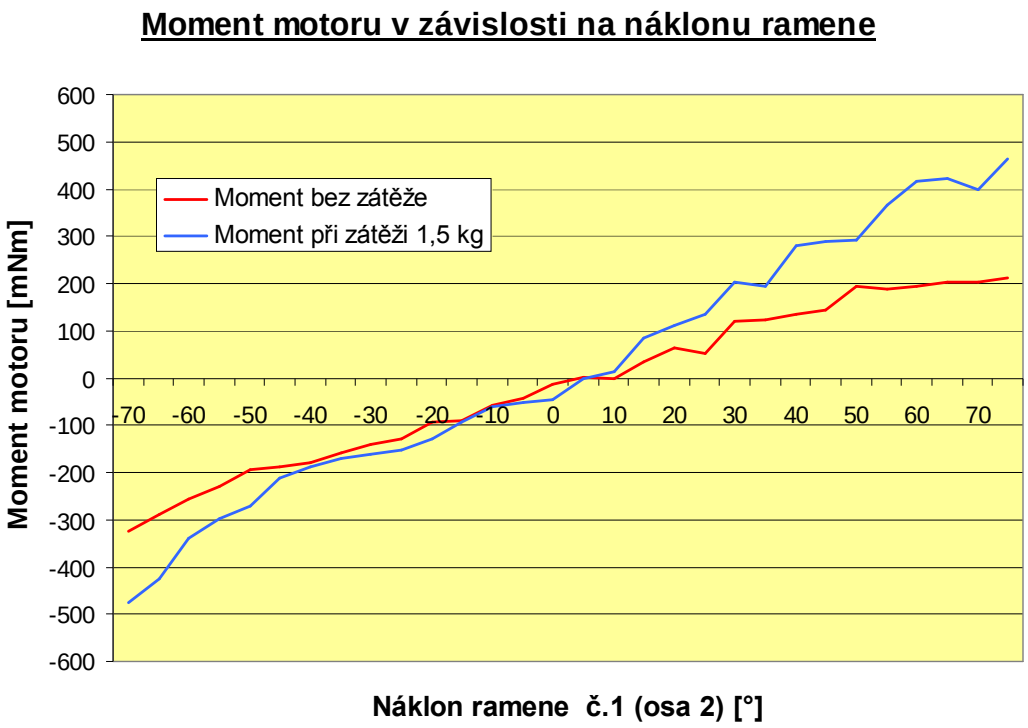
Obr.51 Proud motorem EC60 v celém rozsahu osy

Proud motorem EC60 se pohyboval v hodnotách do 3,8 A bez OM. V případě zatížení příruby OM o hmotnosti 1,5 kg se proud zvýšil na hodnoty do 5,5 A.

Výkyvy křivky jsou způsobeny značnými vůlemi v převodech pohonů, nepřesnostmi ozubených kol a také kmitáním regulátorů v jednotkách EPOS.

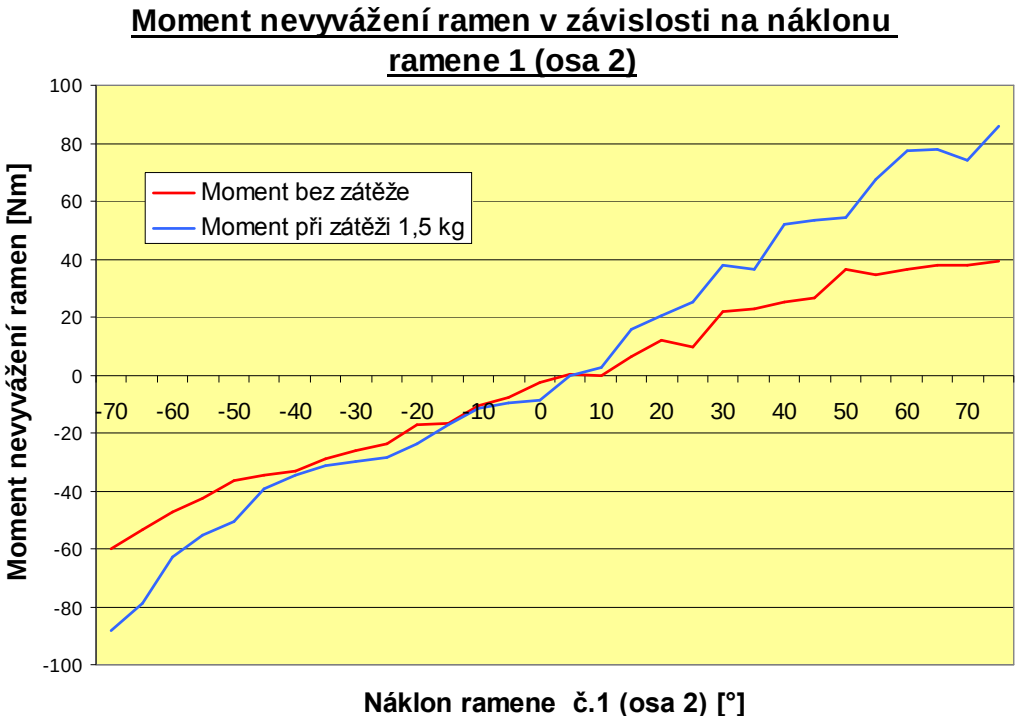
Dle katalogového listu motoru EC60, byla zjištěna jeho konstanta závislosti proudu na momentu. Tato hodnota činí 85 mNm/A.

Změřený graf proudu je tak možno převést na ekvivalentní graf momentu motoru.



Obr.52 Moment na hřídeli motoru EC60

Při přepočtení přes převodový poměr mezi motorem a ramenem získáme přibližný moment nevyvážení ramen manipulátoru.



Obr.53 Moment nevyvážení ramen

Vzhledem k tomu, že regulátory jsou naladěny při jedné poloze ramen, tedy na konstantní zátěž, objevil se problém s kmitáním ramena v ostatních polohách. Nejvíce se tato vlastnost projevila při malých rychlostech pohonu a při velkém vyložení ramen. Proto by bylo vhodné složky regulátoru měnit v závislosti na náklonu ramena. I tato možnost se dá ovšem poměrně snadno realizovat. U jednotek EPOS je možné provádět nastavování složek regulátorů za chodu.

9.9 Seřízení regulátorů pro osu 2

Regulátory jednotky EPOS #2 byly seřízeny pomocí „Autotuningu“ v aplikaci „EPOS User Interface“ ve dvou polohách:

- Pro největší vyložení ramen (q2 = 17° a q3 = 0°)
- Pro ramena svislá (q2 = 90° a q3 = 0°)

Hodnoty regulátorů:

Poloha ramen	**Proudový regulátor PI**		**Rychlostní regulátor PI**		**Polohový regulátor PID**		
	P	**I**	**P**	**I**	**P**	**I**	**D**
Svisle	6835	1473	3607	338	500	40	200
Max. vyložení	6835	1473	4908	338	695	60	200

Tab. 35 Hodnoty složek regulátorů

Po odečtení automaticky nastavených složek regulátorů v obou polohách, byla vypočtena střední hodnota. Tato střední hodnota byla poté vložena do jednotky EPOS.

	Proporcionální složka	Integrační složka	Derivační složka
Regulátor proudu:	P = 6835	I = 1473	-
Regulátor rychlosti	P = 4250	I = 338	-
Regulátor polohy	P = 598	I = 50	D = 200

Tab. 36 Optimální hodnoty složek regulátorů

S takto nastavenými regulátory již manipulátor dosahuje uspokojivých pohybů bez většího kmitání ramen i při nízkých rychlostech. Proto není nutné provádět změnu parametrů regulátorů během pohybu manipulátoru.

10. Využití práce a závěr

Diplomová práce měla za úkol uvést do provozu manipulátor zkonstruovaný na katedře robototechniky.

Výsledkem je provozuschopný manipulátor řízený z PC nebo IPC pomocí aplikace vytvořené k tomuto účelu v rámci diplomové práce. Pro manipulátor byly dle zadání použity mikrokontroléry Maxon EPOS 70/10. Na manipulátoru byla provedena kompletní elektroinstalace řídicích jednotek a jejich potřebného periferního zařízení. Elektrické vedení k pohonům bylo opatřeno ochranou hadicí, aby nedošlo k jeho poškození během pohybu manipulátoru.

Jednotlivé osy manipulátoru byly osazeny mechanickými koncovými dorazy a také koncovými spínači „Positive limit“ a „Negative limit“. Dále byl navržen a sestaven zdroj napájení, který zajišťuje 2 napájecí napětí. Napájení elektromagnetické brzdy motoru EC60 a signál pro koncové spínače zajišťuje běžný spínaný zdroj 24V / 1A. Výkonovou část pro napájení motorů a řídicích jednotek EPOS zajišťuje dvojice spínaných zdrojů o výstupním napětí 58V a výkonu cca 1000W.

Manipulátor je opatřen nouzovým tlačítkem „Emergency STOP“ pro zastavení manipulátoru a zabránění kolize či zranění. Tlačítko má za úkol za jakýchkoliv okolností a stavu řídicích jednotek tyto vypnout a provést aktivaci brzdy.

Manipulátor je připraven pro osazení uchopovací hlavicí, která je předmětem řešení bakalářské práce. Elektroinstalace počítá i s tímto. Proto je do ochranné hadice přidán plochý 10-ti žilový kabel, který je veden od základu až k pohonu příruby . Zde se v budoucnu může provést napojení pohonu efektoru.

Manipulátor by bylo vhodné v budoucnu opravit po mechanické stránce. Bylo by třeba snížit vůle v nábojích pohonů a kuželových ozubeních. Významně by se tak zvýšila jeho přesnost.

Manipulátor bude využíván jako laboratorní úloha určená studentům katedry pro osvojení řízení jednotek EPOS velice často využívaných na katedře robototechniky.

11. Seznam obrázků

12. Seznam tabulek

Tabulka č. 30	Naměřené hodnoty vůlí osy 1
Tabulka č. 31	Naměřené hodnoty vůlí osy 2
Tabulka č. 32	Naměřené hodnoty vůlí osy 3
Tabulka č. 33	Naměřené hodnoty opakovatelné přesnosti osy 1
Tabulka č. 34	Naměřené hodnoty opakovatelné přesnosti osy 2
Tabulka č. 35	Hodnoty složek regulátorů
Tabulka č. 36	Optimální hodnoty složek regulátorů

13. Seznam použité literatury:

[1] JANATA, Pavel. Pokročilé řízení pohonů mechanismů. *MM Průmyslové Spektrum* [Online] Listopad 2001 Dostupné na WWW: < http://www.mmspektrum.com/clanek/pokrocile-rizeni-pohonu-mechanismu >

[2] MOSTÝN, V., SKAŘUPA, J.: *Teorie průmyslových robotů*. Košice 2000, 146 s., ISBN 80-88922-35-6

[3] VOJÁČEK, Antonín. *Motory a jejich řízení s MCU - 1.část - typy motorů.* [Online] 17.Únor 2008 Dostupné na WWW: < http://automatizace.hw.cz/motory-jejich-rizeni-s-mcu-2-cast-spinaci-mustky-jejich-pripojeni-k-mcu >

[4] VOJÁČEK, Antonín. *Motory a jejich řízení s MCU - 2.část - spínací můstky a jejich připojení k MCU*. [Online] 1.Březen 2008 Dostupné na WWW: < http://automatizace.hw.cz/motory-jejich-rizeni-s-mcu-2-cast-spinaci-mustky-jejich-pripojeni-k-mcu >

[5] VOJÁČEK, Antonín. *Motory a jejich řízení s MCU - 3.část - řídicí algoritmy a regulace s MCU*. [Online] 19.Duben 2008 Dostupné na WWW: < http://automatizace.hw.cz/motory-jejich-rizeni-s-mcu-3-cast-ridici-algoritmy-regulace-s-mcu >

[6] *Katalogový list brzdy AB41*. Switzerland: MAXON MOTOR, 2006. , [Online] dostupné na WWW: <http://www.etracker.de/lnkcnt.php?et=pKghaK&url=http://shop.maxonmotor.com/maxon/assets_external/Katalog_neu/Downloads/Katalog_PDF/maxon_accessories/new/Bremse-AB-41-228998_08_EN_310.pdf>

[7] *EPOS 70/10 Hardware reference guide.* Switzerland: MAXON MOTOR, 2006. 39 s.

[8] *EPOS Windows 32-bit DLL*. Switzerland: MAXON MOTOR, 2006. 60 s.

[9] Brož, V. :*Řízení otáček motorů EC s výkony do 400 W*, AUTOMA:Časopis pro automatizační techniku 2007, č.3 [Online] Dostupné na WWW:
< http://www.odbornecasopisy.cz/index.php?id_document=34314>

[11] *Malé stejnosměrné motory MAXON*. UZIMEX Praha. Březen 2004, 55s [Online] , Dostupné na WWW: <http://www.uzimex.cz/soubory/20040315_tat_2004-03.pdf>

[12] MIKULČÍK, A. *EC motor* . Laboratorní úloha, 14.4.2008, [Online] , Dostupné na WWW: <http://jaja.kn.vutbr.cz/~huzlik/EC%20motor.pdf>

[13] *Řízení.* UZIMEX Praha. [Online] Dostupné z WWW:
< http://www.uzimex.cz/Vyrobce/maxon-motor-ag/Rizeni.html>

[14] *EPOS Průvodce* . UZIMEX Praha. Březen 2009. [Online] Dostupné na WWW:
<http://www.uzimex.cz/soubory/20050608_epos_pruvodce_1.2.pdf>

[15] *MAXON RE36 Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW:
<http://test.maxonmotor.com/docsx/Download/catalog_2005/Pdf/05_082_e.pdf>

[16] *MAXON F2260 Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW:
<http://test.maxonmotor.com/docsx/Download/catalog_2005/Pdf/05_096_e.pdf>

[17] *MAXON EC60 Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW:
<http://test.maxonmotor.com/docsx/Download/catalog_2005/Pdf/05_165_e.pdf>

[18] *Motory řady RE.* UZIMEX Praha. [Online]. Dostupné z WWW:
< http://www.uzimex.cz/Vyrobce/maxon-motor-ag/Motory-DC/Rada-RE.html>

[19] *Planetové převodovky.* UZIMEX Praha. [Online]. Dostupné z WWW:
< http://www.uzimex.cz/Vyrobce/maxon-motor-ag/Prevodovky/Planetove-prevodovky.html>

[20] *MAXON GP81A Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW:
<http://www.etracker.de/lnkcnt.php?et=pKghaK&url=http://shop.maxonmotor.com/maxon/assets_external/Katalog_neu/Downloads/Katalog_PDF/maxon_gear/Planetengetriebe/new/GP-81-A-110408_08_EN_250.pdf>

[21] *MAXON GP62A Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW:
<http://www.etracker.de/lnkcnt.php?et=pKghaK&url=http://shop.maxonmotor.com/maxon/assets_external/Katalog_neu/Downloads/Katalog_PDF/maxon_gear/Planetengetriebe/new/GP-62-A-110499_08_EN_249.pdf>

[22] *Motory řady EC.* UZIMEX Praha. [Online]. Dostupné z WWW:
<http://www.uzimex.cz/Vyrobce/maxon-motor-ag/Valcove-motory-EC/Rada-EC.html>

[23] *MAXON GP32 Datasheet.* Switzerland: MAXON MOTOR, 2006 . [Online] Dostupné na WWW: <http://www.etracker.de/lnkcnt.php?et=pKghaK&url=http://shop.maxonmotor.com/maxon/assets_external/Katalog_neu/Downloads/Katalog_PDF/maxon_gear/Planetengetriebe/new/GP-32-C-166930_08_EN_240-241.pdf>

[24] *Motory řady F*. UZIMEX Praha. [Online]. Dostupné z WWW: <http://www.uzimex.cz/Vyrobce/maxon-motor-ag/Motory-DC/Rada-F.html>

[25] *Mikrospínač P-B172C.* GME Praha. [Online]. Dostupné z WWW: <http://www.gme.cz/cz/index.php?product=630-156>

[26] *Spínaný zdroj*. GES Plzeň. [Online]. Dostupné z WWW: <http://www.ges.cz/?page=index&of=2&gcat=XK5N4&inc=detail&gesid=GES07506811>

[27] *PSP-500-48.* MeanWell. United States of America [Online]. Dostupné z WWW: <http://www.meanwell.com/search/PSP-500/PSP-500-spec.pdf>

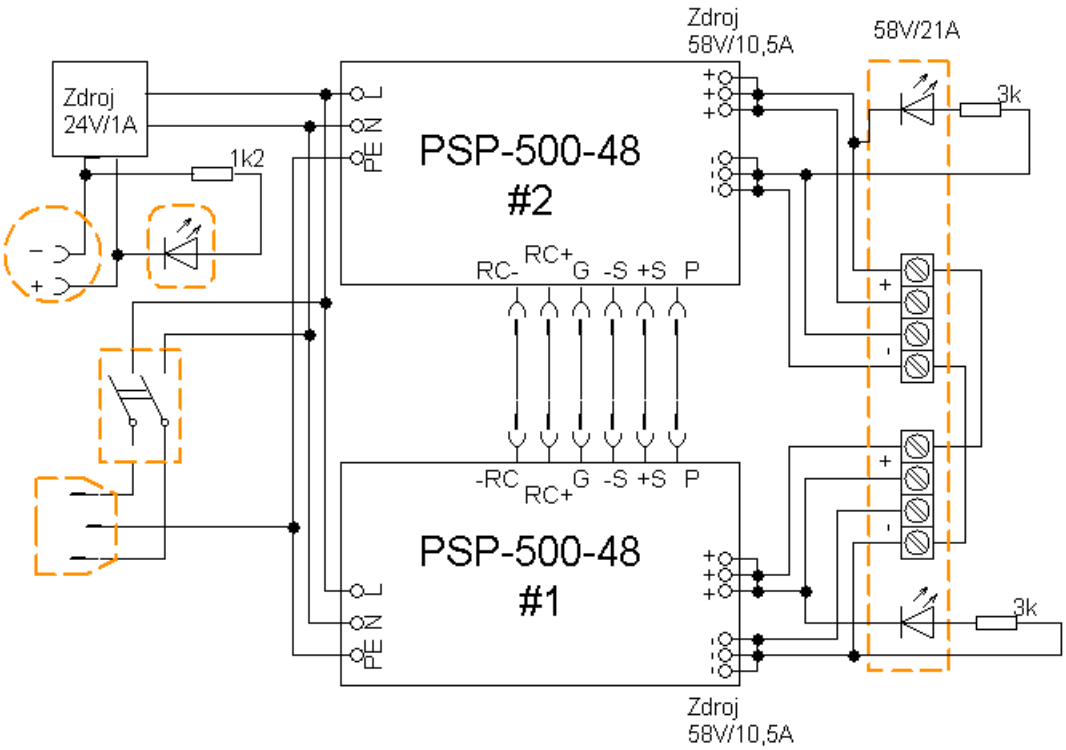
[28] *Elektroinstalační material*.ABB [Online]. Dostupné z WWW: <http://library.abb.com/global/scot/scot209.nsf/veritydisplay/ac3164269c1059a3c125754b00552aa0/$File/1SLC001001D0203.pdf>

14. Přílohy

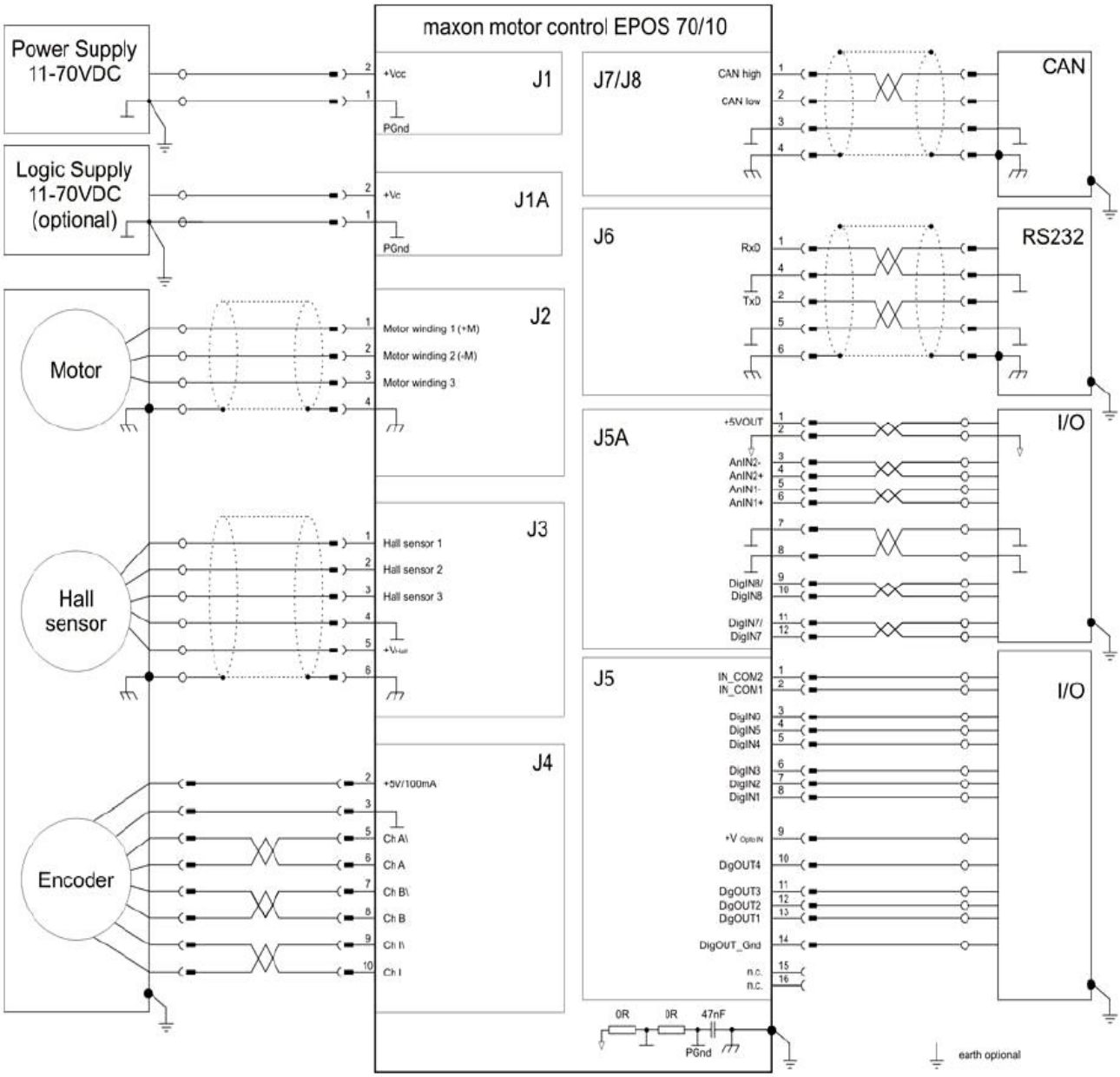
14.1 Schéma zapojení celého manipulátoru

(viz. výkres:EHL006-DP-1 Elektrické zapojení manipulátoru)

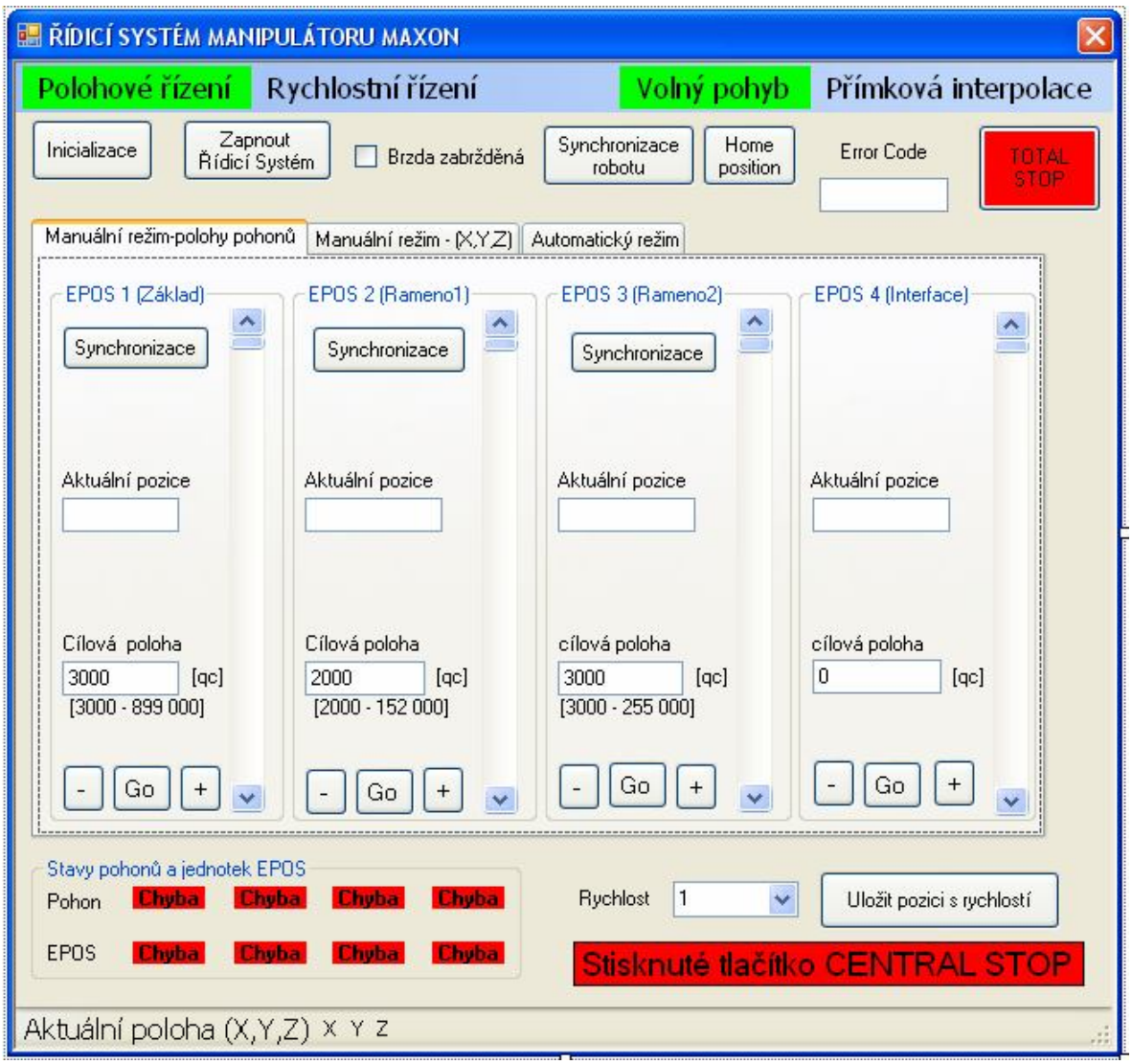
14.2 Schéma zapojení napájecího zdroje



14.3 Schéma řídicí jednotky EPOS 70/10



14.4 Vzhled řídicího programu



14.5 Zdrojový kód řídicího programu

```
Option Explicit On
Public Class Form1
    Dim m_wNodeId As Byte
    Dim dout As Short
    Dim m_lErrorCode As Integer
    Dim m_lImmediately As Integer
    Dim m_lProfileAcceleration, m_lProfileDeceleration As Integer
    Dim m_lProfileVelocity, pbaudrate, ptimeout, amode As Integer
    Dim m_KeyHandle As Object
    Dim m_uMode As Byte
    Dim m_oInitialisation As Boolean
    Dim m_oUpdateActive As Boolean
    Dim oresult As Boolean
    Dim lvalue, pvalue(4), poloha_puls_1, poloha_puls_2,
poloha_puls_3, poloha_puls_4 As Integer
    Dim velvalue As String
    Dim dout1_stat, dout2_stat, dout3_stat, dout4_stat, v1, v2, v3,
v4, n, act1, act2, act3, act4 As Integer
    Dim d4state As Object
    Dim doresult, state(4), state_ep(4), interpol As Boolean
    Dim pdata As Object
    Dim dout_state As Long
    Dim i_node, h, r As Byte
    Dim xi1, xi2, yi1, yi2, zi1, zi2 As Single
    Dim q1, q2, q3, q4, q1t, q2t, q3t, q4t, q1a, q2a, q3a, q4a As
Double
    Dim prevod(4), stoupanix, stoupaniy, stoupaniz, xc, zc, yc, x, y,
z As Double
    Dim R3, R3car, L10, L12, L2, L3, Pcar, P, ALFA, BETA, DELTA, QX As
Double
    Dim p1(50), p2(50), p3(50), p4(50), rychlost(50) As String
    Dim poloha As String
    Dim NbOfBytesWritten, NbOfBytesToRead As Byte
    Dim synch(3) As Boolean
    Dim Response As Object


    Private Function OpenDevice() As Boolean
        'Dim ProtocolStackName As New String
        m_KeyHandle = VCS_OpenDeviceDlg(m_lErrorCode)
        oresult = VCS_GetProtocolStackSettings(m_KeyHandle, pbaudrate,
ptimeout, m_lErrorCode)
        oresult = VCS_SetProtocolStackSettings(m_KeyHandle, 115200,
ptimeout, m_lErrorCode)

        For m_wNodeId = 1 To 4
            If m_KeyHandle Then
                'Deletes the error history
                oresult = VCS_ClearFault(m_KeyHandle, m_wNodeId,
m_lErrorCode)
                If oresult Then
                    'Write Opeational Mode (Profile Position Mode)
                    oresult = VCS_SetOperationMode(m_KeyHandle,
m_wNodeId, 1, m_lErrorCode)
                    If oresult Then
                        'Write Position Profile Objects
```

```
                        oresult = VCS_SetPositionProfile(m_KeyHandle,
m_wNodeId, 1000, 500, 500, m_lErrorCode)
                        If oresult Then
                            'nastavení sinusové rampy rozběhu
                            oresult = VCS_GetObject(m_KeyHandle,
m_wNodeId, &H6086, &H0, amode, 4, NbOfBytesWritten, m_lErrorCode)
                            amode = 1
                            oresult = VCS_SetObject(m_KeyHandle,
m_wNodeId, &H6086, &H0, amode, 4, NbOfBytesWritten, m_lErrorCode)
                            oresult = VCS_Store(m_KeyHandle,
m_wNodeId, m_lErrorCode)
                        End If
                    End If
                End If
            End If

        Next m_wNodeId

        If Not oresult Then
            ShowErrorInformation(m_lErrorCode)
        End If
        OpenDevice = oresult
    End Function


Private Function ShowErrorInformation(ByRef lErrorCode As Integer) As
Boolean
        Const Size_Renamed As Short = 100
        Dim pStrErrorInfo As String
        pStrErrorlabel.Text = (m_lErrorCode)

        If m_lErrorCode = 6 Then
            Response = MsgBox("Chyba v komunikaci s robotem",
MsgBoxStyle.OkOnly, "Chyba")
            Timer1.Stop()
        Else
            Try
                oresult = VCS_GetErrorInfo(m_lErrorCode,
pStrErrorInfo, Size_Renamed)
            Catch
            End Try
        End If
        If oresult = True Then
            Response = MsgBox(pStrErrorInfo, MsgBoxStyle.OkOnly,
"Error")
        Else
            Response = MsgBox("Chyba v komunikaci s robotem",
MsgBoxStyle.OkOnly, "Chyba")
        End If
        ShowErrorInformation = oresult
    End Function


    Private Sub initbt_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles initbt.Click

        If OpenDevice() = True Then
            m_oInitialisation = True
            m_oUpdateActive = True
            Timer1.Enabled = True
```

```
            Response = MsgBox("Má se provést synchronizace robotu??",
vbOKCancel, "Synchronizace")
            If Response = vbOK Then
                Call defhomebt_Click(defhomebt, e)
            Else
                For i = 1 To 3
                    synch(i) = True
                Next
            End If

        Else
            m_oInitialisation = False
            m_oUpdateActive = False
        End If
    End Sub


    Private Sub Form1_FormClosed(ByVal sender As Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles Me.FormClosed
        If MsgBox("Opravdu chete ukončit aplikaci??", vbOKCancel,
"Konec") = MsgBoxResult.Ok Then oresult = RestoreEPOS()

    End Sub


    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        L10 = 228 + 404
        L12 = 103
        L2 = 610
        L3 = 400 + 102
        m_lImmediately = 1
        m_wNodeId = 1
        m_uMode = 1
        prevod(1) = (421824 / 1715) * 2  '246 * 2 (prevodovka *
remenice) = 492
        prevod(2) = (107163 / 1156) * 2 '92,7 * 2 (prevodovka *
kuzelová kola) 185,4
        prevod(3) = (185193 / 1331) '139 (prevodovka)
        prevod(4) = 421824 / 1715 '246 (prevodovka)
        h = 0
        For i = 1 To 3
            synch(i) = False
        Next
    End Sub


    Private Function RestoreEPOS() As Boolean

        If m_oInitialisation = True Then
            oresult = VCS_DigitalOutputConfiguration(m_KeyHandle, 2,
4, 12, False, True, False, m_lErrorCode)
            For m_wNodeId = 1 To 4
                oresult = VCS_SetDisableState(m_KeyHandle, m_wNodeId,
m_lErrorCode)
                oresult = VCS_SetOperationMode(m_KeyHandle, m_wNodeId,
m_uMode, m_lErrorCode)
                If Not oresult Then ShowErrorInformation(m_lErrorCode)
            Next m_wNodeId
        End If
        Timer1.Stop()
```

```
        oresult = VCS_CloseAllDevices(m_lErrorCode)

        RestoreEPOS = oresult
    End Function



    Private Function inverze(ByVal x, ByVal y, ByVal z) As Boolean

        'vektorová inverzní úloha
        R3 = Math.Sqrt(x ^ 2 + y ^ 2)
        R3car = Math.Sqrt(R3 ^ 2 - L12 ^ 2)
        Pcar = Math.Sqrt(R3car ^ 2 + (z - L10) ^ 2)
        P = Math.Sqrt(Pcar ^ 2 + L12 ^ 2)
        If P < (L2 + L3) Then
            ALFA = Math.Acos(-(R3car ^ 2 + (z - L10) ^ 2 - L3 ^ 2 - L2
^ 2) / (2 * L3 * L2))
            q3 = Math.PI - ALFA
            DELTA = Math.Atan((z - L10) / R3car)
            BETA = Math.Acos((L2 ^ 2 - L3 ^ 2 + Pcar ^ 2) / (2 * Pcar
* L2))
            q2 = Math.PI - BETA - DELTA
            QX = Math.Acos(R3car / R3)
            q1 = 1.5 * Math.PI + Math.Atan2(-x, y) - QX

            'přepočet kloubových proměnných q v [rad] na polohy pohonů
v [qc] s korekcí pro posunutou nulovou polohu
            poloha_puls_1 = (1000 * prevod(1) * q1) / Math.PI
            poloha_puls_2 = 152000 - (1000 * prevod(2) * q2 - 65000) /
Math.PI
            poloha_puls_3 = (1000 * prevod(3) * q3 + 378300) / Math.PI
            poloha_puls_4 = (1000 * prevod(4) * q4) / Math.PI

            If poloha_puls_1 < 3000 Or poloha_puls_1 > 899000 Or
poloha_puls_2 < 2000 Or poloha_puls_2 > 152000 Or poloha_puls_2 < 3000
Or poloha_puls_2 > 255000 Then
                MsgBox("Požadovaný bod je mimo pracovní dosah
robotu!!", vbOKOnly, "Error")
                inverze = False
            Else
                inverze = True
            End If
        Else
            MsgBox("Požadovaný bod je mimo pracovní dosah robotu!!",
vbOKOnly, "Error")
            inverze = False
        End If
    End Function


    Private Sub Buttonstart_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Buttonstart.Click
        Dim i As Byte
        If menu_polohove.BackColor = Color.Lime Then

            For i = 1 To 4
                ' profile velocity mode
                oresult = VCS_SetOperationMode(m_KeyHandle, i, 1,
m_lErrorCode)
                If oresult Then
```

```
                    oresult = VCS_SetPositionProfile(m_KeyHandle, i,
rych.Text * prevod(i), prevod(i) * 2, prevod(i) * 2, m_lErrorCode)
                End If
            Next i

            'pohyb do požadované polohy po nedefinované trajektorii
poohovým řízením
            oresult = VCS_MoveToPosition(m_KeyHandle, 1,
poloha_puls_1, True, m_lImmediately, m_lErrorCode)
            oresult = VCS_MoveToPosition(m_KeyHandle, 2,
poloha_puls_2, True, m_lImmediately, m_lErrorCode)
            oresult = VCS_MoveToPosition(m_KeyHandle, 3,
poloha_puls_3, True, m_lImmediately, m_lErrorCode)
            oresult = VCS_MoveToPosition(m_KeyHandle, 4,
poloha_puls_4, True, m_lImmediately, m_lErrorCode)
        End If
         If oresult = True Then
            If Not oresult Then
                oresult = ShowErrorInformation(m_lErrorCode)
            End If
        End If
    End Sub


    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button2.Click
        Timer_interpol.Stop()
        Timer_program.Stop()
        oresult = VCS_DigitalOutputConfiguration(m_KeyHandle, 2, 4,
12, False, True, False, m_lErrorCode)
        For i_node = 1 To 4
            oresult = VCS_SetDisableState(m_KeyHandle, i_node,
m_lErrorCode)
            If oresult Then
                enablebt.Text = "Zapnout Řídicí Systém"
            End If
        Next

    End Sub


    Private Function StopTimer() As Boolean
        Timer1.Stop()
        m_oUpdateActive = False
    End Function

    Private Sub updatestatus()
        Dim dinstat1, doutstat As Short
        Dim proud(4) As Integer
        Dim otemp As String
        'zjisteni stavu nouzového tlacitka EMERGENCY STOP
        oresult = VCS_GetAllDigitalInputs(m_KeyHandle, 2, dinstat1,
m_lErrorCode)
        If dinstat1 < 16 Then
            If stoplabel.Visible = False Then
                stoplabel.Visible = True
            Else
                stoplabel.Visible = False
            End If
        Else
            stoplabel.Visible = False
```

```
        End If

        oresult = VCS_GetAllDigitalOutputs(m_KeyHandle, 2, doutstat,
m_lErrorCode)
        If doutstat = 1 Then
            checkenabled.Checked = True
        Else
            checkenabled.Checked = False
        End If

        'zjistení stavu pohonu
        oresult = VCS_GetMovementState(m_KeyHandle, 1, state(1),
m_lErrorCode)
        oresult = VCS_GetMovementState(m_KeyHandle, 2, state(2),
m_lErrorCode)
        oresult = VCS_GetMovementState(m_KeyHandle, 3, state(3),
m_lErrorCode)
        oresult = VCS_GetMovementState(m_KeyHandle, 4, state(4),
m_lErrorCode)

        If state(1) Then
            stav1.BackColor = Color.Lime
            stav1.Text = "Hotovo"
        Else
            stav1.BackColor = Color.Red
            stav1.Text = "Pracuje"
        End If

        If state(2) Then
            stav2.BackColor = Color.Lime
            stav2.Text = "Hotovo"
        Else
            stav2.BackColor = Color.Red
            stav2.Text = "Pracuje"
        End If

        If state(3) Then
            stav3.BackColor = Color.Lime
            stav3.Text = "Hotovo"
        Else
            stav3.BackColor = Color.Red
            stav3.Text = "Pracuje"
        End If

        If state(4) Then
            stav4.BackColor = Color.Lime
            stav4.Text = "Hotovo"
        Else
            stav4.BackColor = Color.Red
            stav4.Text = "Pracuje"
        End If

        'zjisteni stavu jednotek EPOS
        oresult = VCS_GetEnableState(m_KeyHandle, 1, state_ep(1),
m_lErrorCode)
        oresult = VCS_GetEnableState(m_KeyHandle, 2, state_ep(2),
m_lErrorCode)
        oresult = VCS_GetEnableState(m_KeyHandle, 3, state_ep(3),
m_lErrorCode)
        oresult = VCS_GetEnableState(m_KeyHandle, 4, state_ep(4),
m_lErrorCode)
```

```
        If state_ep(1) Then
            stave1.BackColor = Color.Lime
            stave1.Text = "ON"
        Else
            oresult = VCS_GetFaultState(m_KeyHandle, 1, state_ep(1),
m_lErrorCode)
            If state_ep(1) = False Then
                stave1.BackColor = Color.Yellow
                stave1.Text = "Off"
            Else
                stave1.BackColor = Color.Red
                stave1.Text = "Chyba"
            End If
        End If

        If state_ep(2) Then
            stave2.BackColor = Color.Lime
            stave2.Text = "ON"
        Else
            oresult = VCS_GetFaultState(m_KeyHandle, 2, state_ep(2),
m_lErrorCode)
            If state_ep(2) = False Then
                stave2.BackColor = Color.Yellow
                stave2.Text = "Off"
            Else
                stave2.BackColor = Color.Red
                stave2.Text = "Chyba"
            End If
        End If

        If state_ep(3) Then
            stave3.BackColor = Color.Lime
            stave3.Text = "ON"
        Else
            oresult = VCS_GetFaultState(m_KeyHandle, 3, state_ep(3),
m_lErrorCode)
            If state_ep(3) = False Then
                stave3.BackColor = Color.Yellow
                stave3.Text = "Off"
            Else
                stave3.BackColor = Color.Red
                stave3.Text = "Chyba"
            End If
        End If

        If state_ep(4) Then
            stave4.BackColor = Color.Lime
            stave4.Text = "ON"
        Else
            oresult = VCS_GetFaultState(m_KeyHandle, 4, state_ep(4),
m_lErrorCode)
            If state_ep(4) = False Then
                stave4.BackColor = Color.Yellow
                stave4.Text = "Off"
            Else
                stave4.BackColor = Color.Red
                stave4.Text = "Chyba"
            End If
        End If
```

```
        'zobrazeni aktualni polohy by elf
        If oresult = True Then

            'osa 1
            oresult = VCS_GetPositionIs(m_KeyHandle, 1, lvalue,
m_lErrorCode)
            If oresult = True Then
                actp1.Text = lvalue.ToString
            End If
        Else
            Timer1.Stop()
            otemp = ShowErrorInformation(m_lErrorCode)
            actp1.Text = Str(0)
        End If

        'osa 2
        oresult = VCS_GetPositionIs(m_KeyHandle, 2, lvalue,
m_lErrorCode)
        If oresult = True Then
            actp2.Text = lvalue.ToString
        Else
            Timer1.Stop()
            otemp = ShowErrorInformation(m_lErrorCode)
            actp2.Text = Str(0)
        End If

        'osa(3)
        oresult = VCS_GetPositionIs(m_KeyHandle, 3, lvalue,
m_lErrorCode)
        If oresult = True Then
            actp3.Text = lvalue.ToString
        Else
            Timer1.Stop()
            otemp = ShowErrorInformation(m_lErrorCode)
            actp3.Text = Str(0)
        End If

        'osa(4)
        oresult = VCS_GetPositionIs(m_KeyHandle, 4, lvalue,
m_lErrorCode)
        If oresult = True Then
            actp4.Text = lvalue.ToString
        Else
            Timer1.Stop()
            otemp = ShowErrorInformation(m_lErrorCode)
            actp4.Text = Str(0)
        End If


        'aktuální souřadnice x y z
        xtext.Text = Math.Round(primax(Val(actp1.Text),
Val(actp2.Text), Val(actp3.Text)))
        If 4 - xtext.Text.Length > 0 Then
            For i = 1 To (4 - xtext.Text.Length)
                xtext.Text = " " + xtext.Text
            Next
        End If

        ytext.Text = Math.Round(primay(Val(actp1.Text),
Val(actp2.Text), Val(actp3.Text)))
```

```
        If 4 - ytext.Text.Length > 0 Then
            For i = 1 To (4 - ytext.Text.Length)
                ytext.Text = " " + ytext.Text
            Next
        End If

        ztext.Text = Math.Round(primaz(Val(actp1.Text),
Val(actp2.Text), Val(actp3.Text)))
        If 4 - ztext.Text.Length > 0 Then
            For i = 1 To (4 - ztext.Text.Length)
                ztext.Text = " " + ztext.Text
            Next
        End If


        'zjistení behu programu
        If Timer_interpol.Enabled = True Or Timer_rychlost.Enabled =
True Then
            If prog_stat.BackColor = Color.Green Then
                prog_stat.BackColor = Color.Gold
            Else
                prog_stat.Text = "!!Program běží!!"
                prog_stat.BackColor = Color.Green
            End If
        Else
            prog_stat.Text = "Program stojí"
            prog_stat.BackColor = Color.Red
        End If

    End Sub


    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        updatestatus()
    End Sub


    Private Sub node1_minus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node1_minus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 1, -10000, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub node1_plus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node1_plus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 1, 10000, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub node2_minus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node2_minus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 2, -5200, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub node2_plus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node2_plus.Click
```

```
        oresult = VCS_MoveToPosition(m_KeyHandle, 2, 5200, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub node3_minus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node3_minus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 3, -1000, False,
m_lImmediately, m_lErrorCode)
    End Sub
    Private Sub node3_plus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node3_plus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 3, 1000, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub enable_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles enablebt.Click
        Dim p1value, p2value, p3value, p4value As Integer
        If synch(1) And synch(2) And synch(3) Then
            If enablebt.Text = "Zapnout Řídicí Systém" Then
                For i_node = 1 To 4
                    oresult = VCS_SetOperationMode(m_KeyHandle,
i_node, 1, m_lErrorCode)
                    oresult = VCS_SetPositionProfile(m_KeyHandle,
i_node, 2 * prevod(i_node), 2 * prevod(i_node), 2 * prevod(i_node),
m_lErrorCode)

                    If oresult = False Then Exit For
                Next i_node

                oresult = VCS_SetEnableState(m_KeyHandle, 1,
m_lErrorCode)
                oresult = VCS_SetEnableState(m_KeyHandle, 3,
m_lErrorCode)
                oresult = VCS_SetEnableState(m_KeyHandle, 4,
m_lErrorCode)
                oresult = VCS_SetEnableState(m_KeyHandle, 2,
m_lErrorCode)

                VCS_GetPositionIs(m_KeyHandle, 1, p1value,
m_lErrorCode)
                If p1value < 3000 Then
                    VCS_MoveToPosition(m_KeyHandle, 1, 4000, True,
m_lImmediately, m_lErrorCode)
                    VScrollBar1.Value = 4000
                Else
                    VScrollBar1.Value = p1value
                    targetp1.Text = VScrollBar1.Value
                End If

                oresult = VCS_GetPositionIs(m_KeyHandle, 2, p2value,
m_lErrorCode)
                If p2value < 2000 Then
                    VCS_MoveToPosition(m_KeyHandle, 2, 3000, True,
m_lImmediately, m_lErrorCode)
                    VScrollBar2.Value = 3000
                Else
                    VScrollBar2.Value = p2value
                    targetp2.Text = VScrollBar2.Value
```

```
                End If

                VCS_GetPositionIs(m_KeyHandle, 3, p3value,
m_lErrorCode)
                If p3value < 3000 Then
                    VCS_MoveToPosition(m_KeyHandle, 3, 4000, True,
m_lImmediately, m_lErrorCode)
                    VScrollBar3.Value = 4000
                Else
                    VScrollBar3.Value = p3value
                    targetp3.Text = VScrollBar3.Value
                End If

                VCS_GetPositionIs(m_KeyHandle, 4, p4value,
m_lErrorCode)
                VScrollBar4.Value = p4value
                Targetp4.Text = VScrollBar4.Value

                If oresult = True Then oresult =
VCS_DigitalOutputConfiguration(m_KeyHandle, 2, 4, 12, True, True,
False, m_lErrorCode) 'odbrzdí brzdu na EPOS 2

                If oresult = True Then
                    enablebt.Text = "Vypnout Řídicí Systém"
                Else
                    enablebt.Text = "Zapnout Řídicí Systém"
                End If
            Else
                oresult = VCS_DigitalOutputConfiguration(m_KeyHandle,
2, 4, 12, False, True, False, m_lErrorCode)
                For i_node = 1 To 4
                    oresult = VCS_SetDisableState(m_KeyHandle, i_node,
m_lErrorCode)
                    If oresult = False Then Exit For
                Next i_node
                If oresult = True Then
                    enablebt.Text = "Zapnout Řídicí Systém"
                Else
                    enablebt.Text = "Vypnout Řídicí Systém"
                End If
            End If
        Else
            MsgBox("Robot není synchronizován. Proveďte synchronizaci
pohonů!!", vbOKOnly, "Chyba")
        End If
    End Sub


    Private Sub compinv_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles compinv.Click
        x = Val(targetx.Text)
        y = Val(targety.Text)
        z = Val(targetz.Text)
        'výpocet inverzní ulohy
        If inverze(x, y, z) = True Then

            target1_label.Text = poloha_puls_1.ToString
            target2_label.Text = poloha_puls_2.ToString
            target3_label.Text = poloha_puls_3.ToString
            target4_label.Text = poloha_puls_4.ToString
        End If
```

```
    End Sub


    Private Sub defhomebt_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles defhomebt.Click
        Call Button6_Click(synchro1, e)
        Call Button7_Click(synchro2, e)
        Call Button8_Click(synchro3, e)
    End Sub


    Private Sub node4_plus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node4_plus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 4, 10000, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub gobt1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles gobt1.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 1,
Val(targetp1.Text), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub node4_minus_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles node4_minus.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 4, -10000, False,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub gobt2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles gobt2.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 2,
Val(targetp2.Text), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub gobt3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles gobt3.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 3,
Val(targetp3.Text), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub gobt4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles gobt4.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 4,
Val(Targetp4.Text), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer_interpol.Tick
        Dim act1, act2, act3, act4 As Integer

        oresult = VCS_GetPositionIs(m_KeyHandle, 1, act1,
m_lErrorCode)
        oresult = VCS_GetPositionIs(m_KeyHandle, 2, act2,
m_lErrorCode)
```

```
        oresult = VCS_GetPositionIs(m_KeyHandle, 3, act3,
m_lErrorCode)
        oresult = VCS_GetPositionIs(m_KeyHandle, 4, act4,
m_lErrorCode)

        ' If (Val(targetp1.Text) - act1) < 10000 And
(Val(targetp2.Text) - act2) < 10000 And (Val(targetp3.Text) - act3) <
10000 Then
        'Timer_interpol.Stop()
        ' End If

        'souřadnice startovacího bodu
        xi1 = primax(act1, act2, act3)
        yi1 = primay(act1, act2, act3)
        zi1 = primaz(act1, act2, act3)

        If Math.Abs(Val(xi1) - Val(xi2)) < 2 And Math.Abs(Val(yi1) -
Val(yi2)) < 2 And Math.Abs(Val(zi1) - Val(zi2)) < 2 Then
            Timer_interpol.Stop()
            Exit Sub
        End If

        oresult = rozdil()

        If oresult Then
            xc = xi1 + stoupanix
            yc = yi1 + stoupaniy
            zc = zi1 + stoupaniz

            oresult = inverze(xc, yc, zc)
            If menu_rychlostni.BackColor = Color.Lime Then
                Call Timer_rychlost_Tick(Timer_rychlost, e)

                oresult = VCS_MoveWithVelocity(m_KeyHandle, 1, v1,
m_lErrorCode)
                oresult = VCS_MoveWithVelocity(m_KeyHandle, 2, v2,
m_lErrorCode)
                oresult = VCS_MoveWithVelocity(m_KeyHandle, 3, v3,
m_lErrorCode)
                oresult = VCS_MoveWithVelocity(m_KeyHandle, 4, v4,
m_lErrorCode)

            Else
                If oresult Then
                    oresult = VCS_MoveToPosition(m_KeyHandle, 1,
poloha_puls_1, True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 2,
poloha_puls_2, True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 3,
poloha_puls_3, True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 4,
poloha_puls_4, True, m_lImmediately, m_lErrorCode)
                Else
                    MsgBox("Požadovaný bod je mimo pracovní dosah
robotu!!", vbOKOnly, "Error")
                    Timer_interpol.Stop()
                    Timer_program.Stop()
                End If
            End If
        End If
    End Sub
```

```
    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles savebt.Click
        Dim soubor As IO.StreamWriter
        SaveFileDialog1.ShowDialog()
        Try
            soubor = New IO.StreamWriter(SaveFileDialog1.FileName)
        Catch
        End Try
        For i = 0 To h - 1
            soubor.Write(i.ToString + "   " + p1(i) + "     " + p2(i)
+ "     " + p3(i) + "     " + p4(i) + "     " + rychlost(i) + "   ")
            If i <> (h - 1) Then soubor.Write(vbCrLf)
        Next i
        soubor.Close()

        h = 0
        r = 0
    End Sub


    Private Sub PolohovéŘízeníToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
menu_polohove.Click
        m_uMode = 254
        menu_polohove.BackColor = Color.Lime
        menu_rychlostni.BackColor = Color.Empty
    End Sub


    Private Sub RychlostníŘízeníToolStripMenuItem_Click(ByVal sender
As System.Object, ByVal e As System.EventArgs) Handles
menu_rychlostni.Click
        m_uMode = 254
        menu_polohove.BackColor = Color.Empty
        menu_rychlostni.BackColor = Color.Lime
    End Sub


    Public Function rychlostni() As Boolean
        oresult = inverze(xc, yc, zc)
        If oresult = True Then
            v1 = (targetp1.Text - actp1.Text) /
Val(Timer_interpol.Interval)
            v2 = (targetp2.Text - actp2.Text) /
Val(Timer_interpol.Interval)
            v3 = (targetp3.Text - actp3.Text) /
Val(Timer_interpol.Interval)
            v4 = (Targetp4.Text - actp4.Text) /
Val(Timer_interpol.Interval)

        End If
        ' xc()
        'yc()
        'zc()
        'rychlost = True

        If actp1.Text <> targetp1.Text Then
            oresult = VCS_SetVelocityMust(m_KeyHandle, 1, v1,
m_lErrorCode)
```

```
            oresult = VCS_SetVelocityMust(m_KeyHandle, 2, v2,
m_lErrorCode)
            oresult = VCS_SetVelocityMust(m_KeyHandle, 3, v3,
m_lErrorCode)
            oresult = VCS_SetVelocityMust(m_KeyHandle, 4, v4,
m_lErrorCode)
        End If
    End Function


    Private Sub homebt_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles homebt.Click
        oresult = VCS_MoveToPosition(m_KeyHandle, 1, 891000, True,
m_lImmediately, m_lErrorCode)
        oresult = VCS_MoveToPosition(m_KeyHandle, 2, 72000, True,
m_lImmediately, m_lErrorCode)
        oresult = VCS_MoveToPosition(m_KeyHandle, 3, 248811, True,
m_lImmediately, m_lErrorCode)
        oresult = VCS_MoveToPosition(m_KeyHandle, 4, 0, True,
m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub Button6_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles synchro1.Click
        Dim ohome As Boolean
        Dim msghome As String
        Dim i As Byte
        'Write Opeational Mode (Homing mode)
        oresult = VCS_SetOperationMode(m_KeyHandle, 1, 6,
m_lErrorCode)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 1, 100, 200, 50, -
10000, 10, 891000, m_lErrorCode)

        If MsgBox("!!POZOR!! Osa 1 provede nájezd na koncový spoínač
!!Positive Limit!!", MsgBoxStyle.OkCancel, "Upozornění") =
MsgBoxResult.Ok Then
            oresult = VCS_SetEnableState(m_KeyHandle, 1, m_lErrorCode)
            'nájezd osy na negative limit + index (mode 2)
            ohome = VCS_FindHome(m_KeyHandle, 1, 2, m_lErrorCode)
            If ohome Then synch(1) = True
        End If


        'software limits
        Dim NbOfBytesWritten As Long
        Dim index As Integer
        Dim subIndex As Byte
        Dim Data As Long
        Dim NbOfBytesToWrite As Long

        oresult = VCS_GetObject(m_KeyHandle, 1, index, subIndex, Data,
NbOfBytesToWrite, NbOfBytesWritten, m_lErrorCode)
        oresult = VCS_SetObject(m_KeyHandle, 1, &H607D, &H1, 3000, 4,
NbOfBytesWritten, m_lErrorCode)
        oresult = VCS_SetObject(m_KeyHandle, 1, &H607D, &H2, 899000,
4, NbOfBytesWritten, m_lErrorCode)

        oresult = VCS_Store(m_KeyHandle, 1, m_lErrorCode)
        oresult = VCS_SetDisableState(m_KeyHandle, 1, m_lErrorCode)
```

```
        oresult = VCS_SetOperationMode(m_KeyHandle, 1, 1,
m_lErrorCode)
    End Sub


    Private Sub Button7_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles synchro2.Click
        Dim ohome As Boolean
        Dim msghome As String
        Dim i As Byte
        'Write Opeational Mode (Homing mode)
        oresult = VCS_SetOperationMode(m_KeyHandle, 2, 6,
m_lErrorCode)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 2, 10, 30, 10, -
80000, 10, 72000, m_lErrorCode)

        If MsgBox("!!POZOR!! Osa 2 provede nájezd na koncový spoínač
!!Positive Limit!!", MsgBoxStyle.OkCancel, "Upozornění") =
MsgBoxResult.Ok Then
            oresult = VCS_SetEnableState(m_KeyHandle, 2, m_lErrorCode)
            oresult = VCS_DigitalOutputConfiguration(m_KeyHandle, 2,
4, 12, True, True, False, m_lErrorCode)
            'nájezd osy na negative limit + index (mode 2)
            ohome = VCS_FindHome(m_KeyHandle, 2, 2, m_lErrorCode)
            If ohome Then synch(2) = True

        End If


        'software limits
        Dim index As Integer
        Dim subIndex As Byte
        Dim Data As Long
        Dim NbOfBytesToWrite As Long
        Dim NbOfBytesWritten As Long

        index = &H607D      'software limits
        subIndex = &H1      'Min. Position Limit
        NbOfBytesToWrite = 4
        oresult = VCS_SetObject(m_KeyHandle, 2, &H607D, &H1, 2000, 4,
NbOfBytesWritten, m_lErrorCode)

        subIndex = &H2      'Max. Position Limit
        Data = 152000            ' " limit počtu qc"
        oresult = VCS_SetObject(m_KeyHandle, 2, &H607D, &H2, 152000,
4, NbOfBytesWritten, m_lErrorCode)

        oresult = VCS_Store(m_KeyHandle, 2, m_lErrorCode)
        oresult = VCS_DigitalOutputConfiguration(m_KeyHandle, 2, 4,
12, False, True, False, m_lErrorCode)
        oresult = VCS_SetDisableState(m_KeyHandle, 2, m_lErrorCode)
        oresult = VCS_SetOperationMode(m_KeyHandle, 2, 1,
m_lErrorCode)
    End Sub


    Private Sub Button8_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles synchro3.Click

        Dim ohome As Boolean
        Dim msghome As String
```

```
        Dim i As Byte
        'Write Opeational Mode (Homing mode)
        oresult = VCS_SetOperationMode(m_KeyHandle, 3, 6,
m_lErrorCode)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 3, 10, 30, 10, -
10000, 10, 248800, m_lErrorCode)

        If MsgBox("!!POZOR!! Osa 3 provede nájezd na koncový spoínač
!!Positive Limit!!", MsgBoxStyle.OkCancel, "Upozornění") =
MsgBoxResult.Ok Then
            oresult = VCS_SetEnableState(m_KeyHandle, 3, m_lErrorCode)
            'nájezd osy na negative limit + index (mode 2)
            ohome = VCS_FindHome(m_KeyHandle, 3, 2, m_lErrorCode)
            If ohome Then synch(3) = True

        End If


        'software limits
        Dim index As Integer
        Dim subIndex As Byte
        Dim Data As Long
        Dim NbOfBytesToWrite As Long
        Dim NbOfBytesWritten As Long

        index = &H607D      'software limits
        subIndex = &H1      'Min. Position Limit
        Data = 0            ' " limit počtu qc"
        NbOfBytesToWrite = 4
        oresult = VCS_SetObject(m_KeyHandle, 3, &H607D, &H1, 3000, 4,
NbOfBytesWritten, m_lErrorCode)

        subIndex = &H2      'Max. Position Limit
        Data = 0            ' " limit počtu qc"
        oresult = VCS_SetObject(m_KeyHandle, 3, &H607D, &H2, 255000,
4, NbOfBytesWritten, m_lErrorCode)

        oresult = VCS_Store(m_KeyHandle, 3, m_lErrorCode)
        oresult = VCS_SetDisableState(m_KeyHandle, 3, m_lErrorCode)
        oresult = VCS_SetOperationMode(m_KeyHandle, 3, 1,
m_lErrorCode)
    End Sub


    Private Sub Button9_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles openbt.Click
        OpenFileDialog1.ShowDialog()
        Dim soubor As IO.StreamReader
        Try
            soubor = New IO.StreamReader(OpenFileDialog1.FileName)
            Text_file.Text = soubor.ReadToEnd
            r = 0
        Catch
        End Try
    End Sub


    Private Sub stepbt_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles stepbt.Click
        Dim radky() As String =
Microsoft.VisualBasic.Split(Text_file.Text, vbCrLf)
```

```
        poloha = radky(r)

        'souřadnice cílového bodu
        xi2 = poloha.Substring(5, 4).ToString
        yi2 = poloha.Substring(14, 4).ToString
        zi2 = poloha.Substring(23, 4).ToString

        If inverze(xi2, yi2, zi2) Then

            targetp1.Text = poloha_puls_1.ToString
            targetp2.Text = poloha_puls_2.ToString
            targetp3.Text = poloha_puls_3.ToString
            Targetp4.Text = poloha.Substring(30, 9)
            velvalue = Val(poloha.Substring(40, 3))

            oresult = VCS_GetPositionIs(m_KeyHandle, 1, act1,
m_lErrorCode)
            oresult = VCS_GetPositionIs(m_KeyHandle, 2, act2,
m_lErrorCode)
            oresult = VCS_GetPositionIs(m_KeyHandle, 3, act3,
m_lErrorCode)
            oresult = VCS_GetPositionIs(m_KeyHandle, 4, act4,
m_lErrorCode)


            v1 = ((Val(targetp1.Text) - act1) / (20 / velvalue)) * 60
/ 2000
            v2 = ((Val(targetp2.Text) - act2) / (20 / velvalue)) * 60
/ 2000
            v3 = ((Val(targetp3.Text) - act3) / (20 / velvalue)) * 60
/ 2000
            v4 = ((Val(Targetp4.Text) - act4) / (20 / velvalue)) * 60
/ 2000


            step_nb.Text = r
            If r = radky.Length - 1 Then
                r = 0
            Else
                r += 1
            End If

            If menu_polohove.BackColor = Color.Lime Then
                If interpol = False Then
                    For i = 1 To 4
                        ' profile position mode
                        oresult = VCS_SetOperationMode(m_KeyHandle, i,
1, m_lErrorCode)
                        If oresult Then
                            oresult =
VCS_SetPositionProfile(m_KeyHandle, i, prevod(i) * 2, prevod(i) * 2,
m_lErrorCode)
                        End If
                    Next i
                    oresult = VCS_SetPositionProfile(m_KeyHandle, 1,
v1, Math.Abs(2 * v1), Math.Abs(2 * v1), m_lErrorCode)
                    oresult = VCS_SetPositionProfile(m_KeyHandle, 2,
v2, Math.Abs(2 * v2), Math.Abs(2 * v2), m_lErrorCode)
                    oresult = VCS_SetPositionProfile(m_KeyHandle, 3,
v3, Math.Abs(2 * v3), Math.Abs(2 * v3), m_lErrorCode)
```

```
                    oresult = VCS_SetPositionProfile(m_KeyHandle, 4,
v4, Math.Abs(2 * v4), Math.Abs(2 * v4), m_lErrorCode)

                    oresult = VCS_MoveToPosition(m_KeyHandle, 1,
Val(targetp1.Text), True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 2,
Val(targetp2.Text), True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 3,
Val(targetp3.Text), True, m_lImmediately, m_lErrorCode)
                    oresult = VCS_MoveToPosition(m_KeyHandle, 4,
Val(Targetp4.Text), True, m_lImmediately, m_lErrorCode)

                ElseIf interpol = True Then
                    For i = 1 To 4
                        ' position mode
                        oresult = VCS_SetOperationMode(m_KeyHandle, i,
1, m_lErrorCode)
                        If oresult Then
                            oresult =
VCS_SetPositionProfile(m_KeyHandle, i, prevod(i), prevod(i),
prevod(i), m_lErrorCode)
                        End If
                    Next i
                    Timer_interpol.Start()
                End If
            ElseIf menu_rychlostni.BackColor = Color.Lime Then
                For i = 1 To 4
                    ' profile velocity mode
                    oresult = VCS_SetOperationMode(m_KeyHandle, i, 3,
m_lErrorCode)
                    If oresult Then
                        oresult = VCS_SetVelocityProfile(m_KeyHandle,
i, prevod(i), prevod(i), m_lErrorCode)
                    End If
                Next i

                If interpol = False Then
                    Timer_rychlost.Start()

                ElseIf interpol = True Then
                    Timer_interpol.Start()
                End If
            End If
        End If
    End Sub


    Private Sub Button3_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button3.Click
        If actp1.Text <> "" Then
            p1(h) = xtext.Text
            p2(h) = ytext.Text
            p3(h) = ztext.Text
            p4(h) = actp4.Text
            If actp4.TextLength < 6 Then
                For i = actp4.TextLength To 5
                    p4(h) = " " + p4(h)
                Next
            End If
            rychlost(h) = rych.Text
```

```
            If h = 0 Then
                Text_file.AppendText(h.ToString + "   " +
p1(h).ToString + "     " + p2(h).ToString + "     " + p3(h).ToString +
"     " + p4(h).ToString + "     " + rychlost(h).ToString)
            Else
                Text_file.AppendText(vbCrLf + h.ToString() + "   " +
p1(h).ToString + "     " + p2(h).ToString + "     " + p3(h).ToString +
"     " + p4(h).ToString + "     " + rychlost(h).ToString)
            End If
            h += 1
        Else
            MsgBox("!!POZOR!! Robot nekomunikuje. Proveďte
inicializaci!!", vbOKOnly, "Upozornění")
        End If
    End Sub


    Private Sub VScrollBar1_MouseCaptureChanged(ByVal sender As
Object, ByVal e As System.EventArgs) Handles
VScrollBar1.MouseCaptureChanged
        oresult = VCS_MoveToPosition(m_KeyHandle, 1,
Val(VScrollBar1.Value), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub VScrollBar3_MouseCaptureChanged(ByVal sender As
Object, ByVal e As System.EventArgs) Handles
VScrollBar3.MouseCaptureChanged
        oresult = VCS_MoveToPosition(m_KeyHandle, 3,
Val(VScrollBar3.Value), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub VScrollBar4_MouseCaptureChanged(ByVal sender As
Object, ByVal e As System.EventArgs) Handles
VScrollBar4.MouseCaptureChanged
        oresult = VCS_MoveToPosition(m_KeyHandle, 4,
Val(VScrollBar4.Value), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub VScrollBar2_MouseCaptureChanged(ByVal sender As
Object, ByVal e As System.EventArgs) Handles
VScrollBar2.MouseCaptureChanged
        oresult = VCS_MoveToPosition(m_KeyHandle, 2,
Val(VScrollBar2.Value), True, m_lImmediately, m_lErrorCode)
    End Sub


    Private Sub Timer_rychlost_Tick(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Timer_rychlost.Tick
        oresult = VCS_GetPositionIs(m_KeyHandle, 1, act1,
m_lErrorCode)
        oresult = VCS_GetPositionIs(m_KeyHandle, 2, act2,
m_lErrorCode)
        oresult = VCS_GetPositionIs(m_KeyHandle, 3, act3,
m_lErrorCode)
        oresult = VCS_GetPositionIs(m_KeyHandle, 4, act4,
m_lErrorCode)

        'souřadnice startovacího bodu
```

```
        xi1 = primax(act1, act2, act3)
        yi1 = primay(act1, act2, act3)
        zi1 = primaz(act1, act2, act3)

        'souřadnice cílového bodu
        xi2 = primax(Val(targetp1.Text), Val(targetp2.Text),
Val(targetp3.Text))
        yi2 = primay(Val(targetp1.Text), Val(targetp2.Text),
Val(targetp3.Text))
        zi2 = primaz(Val(targetp1.Text), Val(targetp2.Text),
Val(targetp3.Text))


        If Math.Abs(Val(targetp1.Text) - act1) > 10000 Then
            v1 = ((Val(targetp1.Text) - act1) / (20 / velvalue)) * 60
/ 2000 + (prevod(1) / 50) * velvalue
        ElseIf Math.Abs(Val(targetp1.Text) - act1) > 5000 Then
            v1 = ((Val(targetp1.Text) - act1) / (20 / velvalue)) * 60
/ 2000
        Else
            v1 = 0
        End If

        If Math.Abs(Val(targetp2.Text) - act2) > 10000 Then
            v2 = ((Val(targetp2.Text) - act2) / (20 / velvalue)) * 60
/ 2000 + (prevod(2) / 50) * velvalue
        ElseIf Math.Abs(Val(targetp2.Text) - act2) > 5000 Then
            v2 = ((Val(targetp2.Text) - act2) / (20 / velvalue)) * 60
/ 2000
        Else
            v2 = 0
        End If

        If Math.Abs(Val(targetp3.Text) - act3) > 10000 Then
            v3 = ((Val(targetp3.Text) - act3) / (20 / velvalue)) * 60
/ 2000 + (prevod(3) / 50) * velvalue
        ElseIf Math.Abs(Val(targetp3.Text) - act3) > 5000 Then
            v3 = ((Val(targetp3.Text) - act3) / (20 / velvalue)) * 60
/ 2000
        Else
            v3 = 0
        End If

        If Math.Abs(Val(Targetp4.Text) - act4) > 10000 Then
            v4 = ((Val(Targetp4.Text) - act4) / (20 / velvalue)) * 60
/ 2000 + (prevod(4) / 50) * velvalue
        ElseIf Math.Abs(Val(Targetp4.Text) - act4) > 5000 Then
            v4 = ((Val(Targetp4.Text) - act4) / (20 / velvalue)) * 60
/ 2000
        Else
            v4 = 0
        End If

        'pohyb do požadované polohy po nedefinované trajektorii
rychlostním řízením
        If menu_rychlostni.BackColor = Color.Lime And
menu_volny.BackColor = Color.Lime Then
            oresult = VCS_MoveWithVelocity(m_KeyHandle, 1, v1,
m_lErrorCode)
            oresult = VCS_MoveWithVelocity(m_KeyHandle, 2, v2,
m_lErrorCode)
```

```
            oresult = VCS_MoveWithVelocity(m_KeyHandle, 3, v3,
m_lErrorCode)
            oresult = VCS_MoveWithVelocity(m_KeyHandle, 4, v4,
m_lErrorCode)
        End If
    End Sub


    Private Sub ToolStripMenuItem1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles menu_volny.Click
        menu_volny.BackColor = Color.Lime
        menu_interpolace.BackColor = Color.Empty
        interpol = False
        For i = 1 To 4
            oresult = VCS_SetOperationMode(m_KeyHandle, i, 1,
m_lErrorCode)
        Next i
    End Sub


    Private Sub menu_interpolace_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles menu_interpolace.Click
        menu_volny.BackColor = Color.Empty
        menu_interpolace.BackColor = Color.Lime
        interpol = True
        For i = 1 To 4
            oresult = VCS_SetOperationMode(m_KeyHandle, i, 255,
m_lErrorCode)
        Next i
    End Sub


    Private Sub clearbt_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles clearbt.Click
        Text_file.Clear()
        r = 0
        h = 0
    End Sub


    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles stopprgbt.Click
        Timer_interpol.Stop()
        Timer_rychlost.Stop()
        Timer_program.Stop()
        For I = 1 To 4
            VCS_HaltVelocityMovement(m_KeyHandle, I, m_lErrorCode)
        Next I
    End Sub


    Public Function rozdil() As Boolean
        Dim rozdilx, rozdily, rozdilz, rozdilmax As Integer
        Dim krokx, kroky, krokz As Single
        rozdilx = (xi2 - xi1)
        rozdily = (yi2 - yi1)
        rozdilz = (zi2 - zi1)

        If rozdilx = 0 Then
            If rozdily = 0 Then
                If rozdilz = 0 Then
```

```
                    rozdil = False
                    Exit Function
                End If
            End If
        End If
        rozdilmax = Math.Max(Math.Abs(rozdilx), Math.Abs(rozdily))
        rozdilmax = Math.Max(Math.Abs(rozdilmax), Math.Abs(rozdilz))
        If rozdilx > 0 Then krokx = 3
        If rozdilx < 0 Then krokx = -3
        If rozdilx = 0 Then krokx = 0
        If rozdily > 0 Then kroky = 3
        If rozdily < 0 Then kroky = -3
        If rozdily = 0 Then kroky = 0
        If rozdilz > 0 Then krokz = 3
        If rozdilz < 0 Then krokz = -3
        If rozdilz = 0 Then krokz = 0
        If rozdilmax = 0 Then Exit Function
        If rozdilmax = rozdilx Or rozdilmax = -rozdilx Then
            stoupanix = krokx
            stoupaniy = kroky * (Math.Abs(rozdily) / rozdilmax)
            stoupaniz = krokz * (Math.Abs(rozdilz) / rozdilmax)
        ElseIf rozdilmax = rozdily Or rozdilmax = -rozdily Then
            stoupaniy = kroky
            stoupanix = krokx * (Math.Abs(rozdilx) / rozdilmax)
            stoupaniz = krokz * (Math.Abs(rozdilz) / rozdilmax)
        ElseIf rozdilmax = rozdilz Or rozdilmax = -rozdilz Then
            stoupaniz = krokz
            stoupanix = krokx * (Math.Abs(rozdilx) / rozdilmax)
            stoupaniy = kroky * (Math.Abs(rozdily) / rozdilmax)
        End If
        rozdil = True
    End Function


    Private Sub VScrollBar1_Scroll(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.ScrollEventArgs) Handles
VScrollBar1.Scroll
        targetp1.Text = VScrollBar1.Value
    End Sub


    Private Sub VScrollBar2_Scroll(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.ScrollEventArgs) Handles
VScrollBar2.Scroll
        targetp2.Text = VScrollBar2.Value
    End Sub


    Private Sub VScrollBar3_Scroll(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.ScrollEventArgs) Handles
VScrollBar3.Scroll
        targetp3.Text = VScrollBar3.Value
    End Sub

    Private Sub VScrollBar4_Scroll(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.ScrollEventArgs) Handles
VScrollBar4.Scroll
        Targetp4.Text = VScrollBar4.Value
    End Sub
```

```
    Sub pokus()
        Dim ohome As Boolean
        Dim msghome As String
        'Write Opeational Mode (Homing mode)
        For i = 1 To 4
            oresult = VCS_SetOperationMode(m_KeyHandle, i, 6,
m_lErrorCode)
        Next i      '(handle,node,zrychlení, rychlost switch, rychlost
index,home offset,current treshold,home position)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 1, 100, 100, 50,
0, 10, 0, m_lErrorCode)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 2, 100, 50, 20, 0,
10, 0, m_lErrorCode)
        ohome = VCS_SetHomingParameter(m_KeyHandle, 3, 100, 100, 20,
0, 10, 0, m_lErrorCode)
        'ohome = VCS_SetHomingParameter(m_KeyHandle, 4, 100, 100, 100,
0, 10, 0, m_lErrorCode)
        MsgBox("!!POZOR!! Robot provede nájezd na koncové spoínače!!",
vbOKOnly, "Error")

        'nájezd osy na negative limit (mode 17)
        ohome = VCS_FindHome(m_KeyHandle, 1, 17, m_lErrorCode)
        If ohome = True Then
            msghome = MsgBox("Negative limit osy 1 nalezen", vbOKOnly,
"Homing mode")
            If msghome = MsgBoxResult.Ok Then
                ohome = VCS_FindHome(m_KeyHandle, 2, 17, m_lErrorCode)
                If ohome = True Then
                    msghome = MsgBox("Negative limit osy 2 nalezen",
vbOKOnly, "Homing mode")
                    If msghome = MsgBoxResult.Ok Then
                        ohome = VCS_FindHome(m_KeyHandle, 3, 17,
m_lErrorCode)
                        MsgBox("Koncový spínač osy 3 nalezen",
vbOKOnly, "Homing mode")
                    Else
                        msghome = MsgBox("Koncový spínač osy 3
nenalezen", vbOKOnly, "Homing mode")
                    End If
                Else
                    msghome = MsgBox("Koncový spínač osy 2 nenalezen",
vbOKOnly, "Homing mode")
                End If
            Else
                msghome = MsgBox("Koncový spínač osy 1 nenalezen",
vbOKOnly, "Homing mode")
            End If
        End If
    End Sub


    Public Function primax(ByVal pos1 As Integer, ByVal pos2 As
Integer, ByVal pos3 As Integer) As Single
        q1 = (Val(pos1) * Math.PI) / (1000 * prevod(1))
        q2 = (Math.PI - (17 * Math.PI / 180) - (Val(pos2) * Math.PI) /
(1000 * prevod(2)))  'správně!!!   +17 stupne přesně
        q3 = (Val(pos3) * Math.PI) / (1000 * prevod(3)) - (378.3 /
(prevod(3)))  'správně!!!   -138,24 stupne
        primax = (Math.Cos(q1) * Math.Cos(q2) * L3 * Math.Cos(q3) -
Math.Cos(q1) * Math.Sin(q2) * L3 * Math.Sin(q3) + Math.Cos(q1) * L2 *
Math.Cos(q2) + Math.Sin(q1) * L12)
```

```
    End Function


    Public Function primay(ByVal pos1 As Integer, ByVal pos2 As
Integer, ByVal pos3 As Integer) As Single
        q1 = (Val(pos1) * Math.PI) / (1000 * prevod(1))
        q2 = (Math.PI - (17 * Math.PI / 180) - (Val(pos2) * Math.PI) /
(1000 * prevod(2)))  'správně!!!   +17 stupne přesně
        q3 = (Val(pos3) * Math.PI) / (1000 * prevod(3)) - (378.3 /
(prevod(3)))  'správně!!!   -138,24 stupne
        primay = (Math.Sin(q1) * Math.Cos(q2) * L3 * Math.Cos(q3) -
Math.Sin(q1) * Math.Sin(q2) * L3 * Math.Sin(q3) + Math.Sin(q1) * L2 *
Math.Cos(q2) - Math.Cos(q1) * L12)
    End Function


    Public Function primaz(ByVal pos1 As Integer, ByVal pos2 As
Integer, ByVal pos3 As Integer) As Single
        q2 = (Math.PI - (17 * Math.PI / 180) - (Val(actp2.Text) *
Math.PI) / (1000 * prevod(2)))  'správně!!!   +17 stupne přesně
        q3 = (Val(actp3.Text) * Math.PI) / (1000 * prevod(3)) - (378.3
/ (prevod(3)))  'správně!!!   -138,24 stupne
        primaz = (Math.Sin(q2) * L3 * Math.Cos(q3) + Math.Cos(q2) * L3
* Math.Sin(q3) + L2 * Math.Sin(q2) + L10)
    End Function


    Private Sub startprgbt_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles startprgbt.Click
        Timer_program.Start()
    End Sub


    Private Sub Timer_program_Tick(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Timer_program.Tick
        If r = 0 Then
            stepbt_Click(stepbt, e)
        elseIf Math.Abs(Val(xtext.Text) - Val(xi2)) < 3 And
Math.Abs(Val(ytext.Text) - Val(yi2)) < 3 And Math.Abs(Val(ztext.Text)
- Val(zi2)) < 3 Then
            stepbt_Click(stepbt, e)
        End If
    End Sub


    Private Sub pausebt_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles pausebt.Click
        If menu_polohove.BackColor = Color.Lime Then
            If Timer_interpol.Enabled Then
                Timer_interpol.Stop()
                Timer_program.Stop()
                pausebt.Text = "Pokračovat v programu"
            Else
                Timer_interpol.Start()
                Timer_program.Start()
                pausebt.Text = "Pozastavit program"
            End If

        ElseIf menu_rychlostni.BackColor = Color.Lime Then
            If Timer_rychlost.Enabled Then
                Timer_rychlost.Stop()
```

```
                Timer_program.Stop()
                pausebt.Text = "Pokračovat v programu"
            Else
                Timer_rychlost.Start()
                Timer_rychlost.Start()
                pausebt.Text = "Pozastavit program"
            End If
        End If
    End Sub
End Class
```